Renesas RA Family

Quick Start Guide: Modbus TCP

Introduction

This document is a quick start guide for evaluating Modbus® communication with the RA microcomputer evaluation board.
Modbus protocol is a communication protocol developed by Modicon Inc. (Schneider Electric SA.) for programmable logic controllers (PLCs), and its specifications are open to the public.
For details, refer to the protocol specifications (PI-MBUS-300 Rev.J).

Target Device

RA6M3, RA6M4, RA6M5
RA8D1, RA8M1, RA8T1
RA8T2, RA8D2, RA8P1, RA8M2

Supported Evaluation Boards

EK-RA6M3, EK-RA6M4, EK-RA6M5
EK-RA8D1, EK-RA8M1, MCK-RA8T1
MCK-RA8T2, EK-RA8D2, EK-RA8P1, EK-RA8M2

Contents

1. Overview

This document describes the Modbus protocol stack that operates on the RA evaluation board. It provides an overview of its functions and a Modbus sample application for developing and implementing applications using the protocol stack.

This sample program package supports the Ethernet-based Modbus TCP protocol.

This Quick Start Guide provides:

- Sample program package configuration
- Operating environment requirements
- Evaluation board connection setup
- Procedure for creating, modifying, and building the Modbus sample project using the Flexible Software Package (FSP) and e^2 studio Integrated Development Environment (IDE).
- Instructions for connecting with a client and operating a simple demo.

1.1 Abbreviations/Definitions

Table 1.1 Abbreviations/Definitions

Index	Abbreviations /Definitions	Description
1	IP	Internet Protocol
2	TCP	Transmission Control Protocol
3	USB	Universal Serial Bus
4	PC	Personal Computer
5	SW	Switch
6	EWARM	Embedded Workbench® for Arm
7	LED	Light Emitting Diode
8	Wireshark	Free packet capture tool to check packets flowing on LAN

1.2 Reference

Technical information about Modbus is available through the Modbus organization site, and information about the RA evaluation board is available through Renesas.

Table 1.2 Technical Inputs

Index	Technical Inputs
1	Modicon Modbus Protocol Reference Guide Rev.J / PI_MBUS_300.pdf
2	Modbus Application Protocol Specification V1.1b3 / Modbus_Application_Protocol_V1_1b3.pdf
3	Evaluation Kit for RA6M3 Microcontroller Group EK-RA6M3 Quick Start Guide / r20qs0011euxxxx
4	Evaluation Kit for RA6M3 Microcontroller Group EK-RA6M3 v1 User's Manual / r20ut4623euxxxx
5	Evaluation Kit for RA6M4 Microcontroller Group EK-RA6M4 Quick Start Guide / r20qs0016egxxxx
6	Evaluation Kit for RA6M4 Microcontroller Group EK-RA6M4 v1 User's Manual / r20ut4836egxxxx
7	Evaluation Kit for RA6M5 Microcontroller Group EK-RA6M5 Quick Start Guide / r20qs0021egxxxx
8	Evaluation Kit for RA6M5 Microcontroller Group EK-RA6M5 v1 User's Manual / r20ut4829egxxxx
9	Evaluation Kit for RA8D1 Microcontroller Group EK-RA8D1 Quick Start Guide / r20qs0065egxxxx
10	Evaluation Kit for RA8D1 Microcontroller Group EK-RA8D1 v1 User's Manual / r20ut5205egxxxx
11	Evaluation Kit for RA8M1 Microcontroller Group EK-RA8M1 Quick Start Guide / r20qs0035egxxxx
12	Evaluation Kit for RA8M1 Microcontroller Group EK-RA8M1 v1 User's Manual / r20ut5149egxxxx
13	MCK-RA8T1 Quick Start Guide / r12qs0067ejxxxx
14	MCK-RA8T1 User's Manual / r12uz0133ejxxxx
15	MCK-RA8T2 Quick Start Guide / r12qs0088ejxxxx
16	MCK-RA8T2 User's Manual / r12uz0172ejxxxx
17	Evaluation Kit for RA8D2 Microcontroller Group EK-RA8D2 Quick Start Guide / r20qs0077egxxxx
18	Evaluation Kit for RA8D2 Microcontroller Group EK-RA8D2 v1 User's Manual / r20ut5523egxxxx
19	Evaluation Kit for RA8P1 Microcontroller Group EK-RA8P1 Quick Start Guide / r20qs0051egxxxx
20	Evaluation Kit for RA8P1 Microcontroller Group EK-RA8P1 v1 User's Manual / r20ut5309egxxxx
21	Evaluation Kit for RA8M2 Microcontroller Group EK-RA8M2 Quick Start Guide / r20qs0069egxxxx
22	Evaluation Kit for RA8M2 Microcontroller Group EK-RA8M2 v1 User's Manual / r20ut5451egxxxx

2. Features

The Modbus protocol stack for the RA evaluation board enables quick, easy development of Modbus TCP applications.

Modbus function codes supported by the initialization API are also specified. The following nine function codes can be implemented in this stack:

- 1(0x01) – Read coils
- 2(0x02) – Read discrete input
- 3(0x03) – Read holding registers
- 4(0x04) – Read input registers
- 5(0x05) – Write single coil
- 6(0x06) – Write single register
- 15(0x0F) – Write multiple coils
- 16(0x10) – Write multiple registers
- 23(0x17) – Read/Write multiple registers

For more information about Modbus, refer to the following site:
http://www.modbus.org

Note: The version number may differ depending on the update. Refer to the latest manual.

3. Sample Program Package Configuration

This sample program package consists of three components:

- Modbus sample project using Modbus protocol stack.
- Modbus sample application using Modbus protocol stack.
- Modbus sample demo application

3.1 Modbus Sample Project

- Modbus_TCP / project / EK-RA6M5
  - This folder contains a folder called “RA_Modbus” which includes a Modbus sample project for EK-RA6M5 boards that uses the Modbus protocol stack.
  - The sample project is created for GCC compilers. If you want to use another compiler, refer to section **"6.2 Creation of new Modbus Project"** and create another project.
  - The sample project is created for EK-RA6M5. If you want to run it on EK-RA6M3, EK-RA6M4, EK-RA8D1, EK-RA8M1, MCK-RA8T1, refer to step (4) of "**6.1 Import Modbus Sample Project**" and modify the project with respect to different boards.
- Modbus_TCP / project / MCK-RA8T2
- Modbus_TCP / project / EK-RA8D2
- Modbus_TCP / project / EK-RA8P1
- Modbus_TCP / project / EK-RA8M2
  - These folders contain a folder called “RA_Modbus” which includes a Modbus sample project for MCK-RA8T2 / EK-RA8D2 / EK-RA8P1 / EK-RA8M2 boards that uses the Modbus protocol stack.
  - The sample projects are created for GCC compilers. If you want to use another compiler, refer to section "**6.2 Creation of new Modbus Project**" and create another project.

3.2 Modbus Sample Application

- Modbus_TCP / src / modbus_func.c, modbus_user.c, new_thread0_entry.c
  - User can register their own implementations of Modbus function codes in the Modbus protocol stack.
  - The code in this directory provides examples of the Modbus protocol stack initialization process and the Modbus function codes processing using the Modbus protocol stack API.

3.3 Modbus Demo Application

- Modbus_tool / ModbusDemoApplication.exe
  - This executable file is a Modbus demo application used for Modbus communication. It can be used to demonstrate the operation of the Modbus sample application.

4. Operating Environment Requirements

The sample program package described in this manual runs in the following environment.

Table 4.1 Operating environment

Item	Description
Board	RA Evaluation Board
Operating Voltage	3.3 V
Integrated Development Environment	IAR Systems - IAR Embedded Workbench® for Arm Version 9.70.1.13552 or later Renesas Electronics - e² studio 2025-10 or later - Renesas RA Smart Configurator 2025-10 or later
Toolchain	IAR Embedded Workbench for Arm - IAR C/C++ Compiler for Arm 9.70.1.475 (9.70.1.475) or later e² Studio - GCC Arm Embedded (13.2.1.arm-13-7) or later - LLVM for Arm (18.1.3) or later - Arm Compiler 6.23 or later
MCU Software Package	FSP (Flexible Software Package) v6.2.0 or above
Emulator	J-LINK OB
Communications Protocol	Modbus TCP
Client Tool	ModbusDemoApplication.exe: Modbus Demo Application

5. Evaluation Board Connection Setup

Connect the PC to a supported evaluation board. Power is supplied by connecting a USB micro-B cable to the board. For Modbus TCP communication, use an RJ45 connector and connect to the PC with a LAN cable.
The connection setting diagrams for the EK RA8M1 evaluation board, the MCK-RA8T1 board and the MCK-RA8T2 board are shown below.

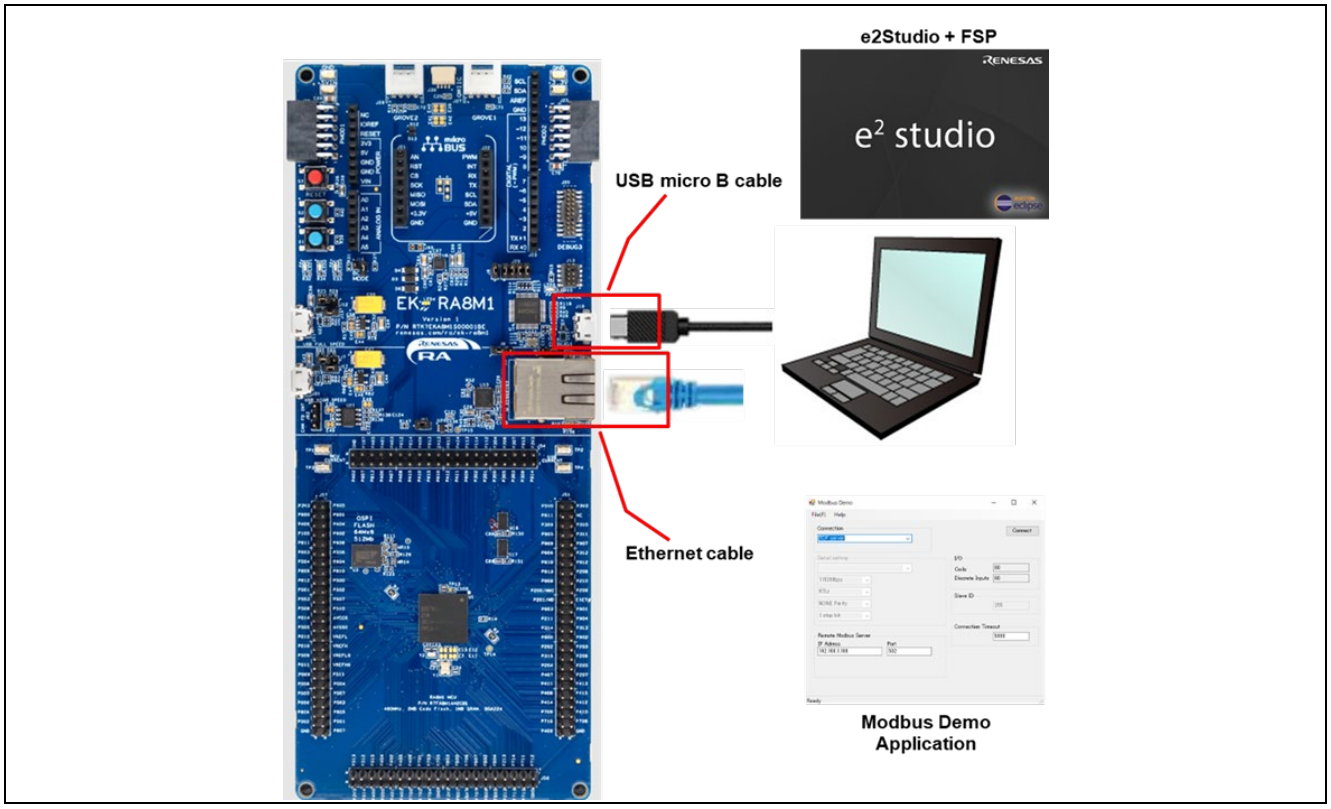


Figure 5.1. EK evaluation board connection setup for RA (Example: EK-RA8M1)

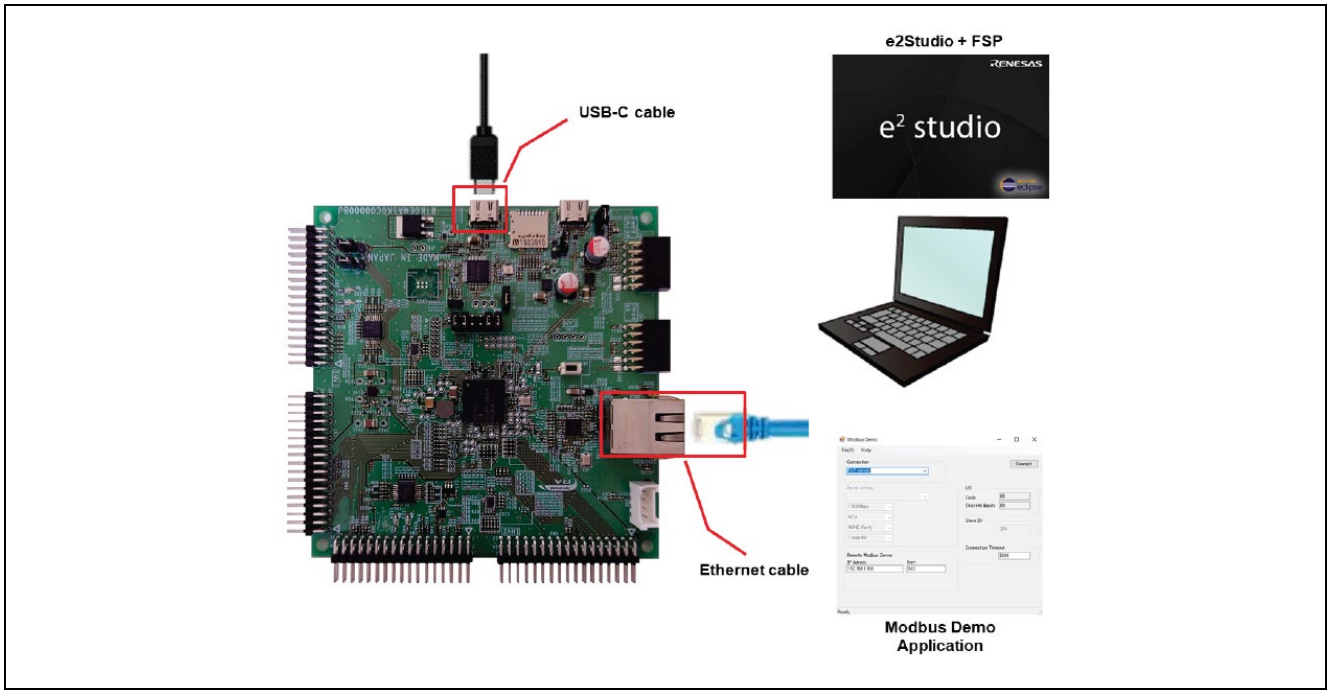


Figure 5.2. MCK-RA8T1 board connection setup

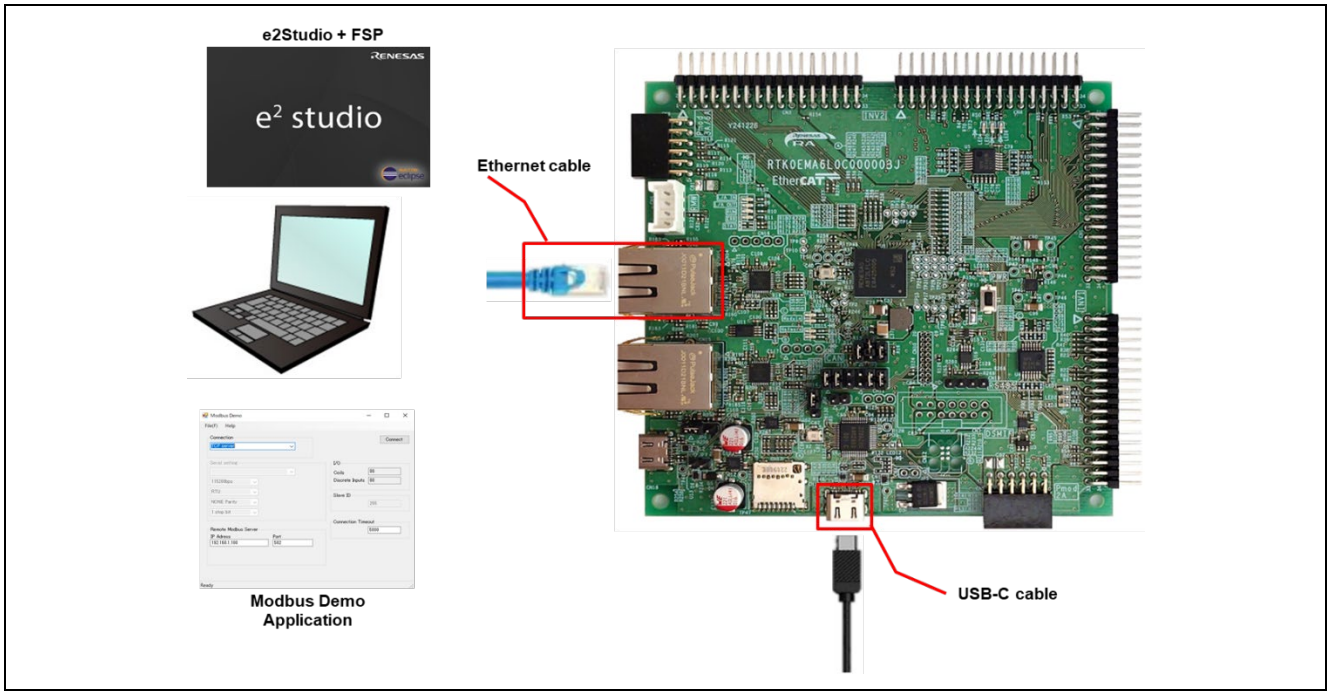


Figure 5.3. MCK-RA8T2 board connection setup

6. Setting Up the Modbus Sample Project

This section describes the procedure for importing/creating a Modbus sample project. Before this, read section "**4. Operating Environment Requirements**" first and complete the installation of the tools.

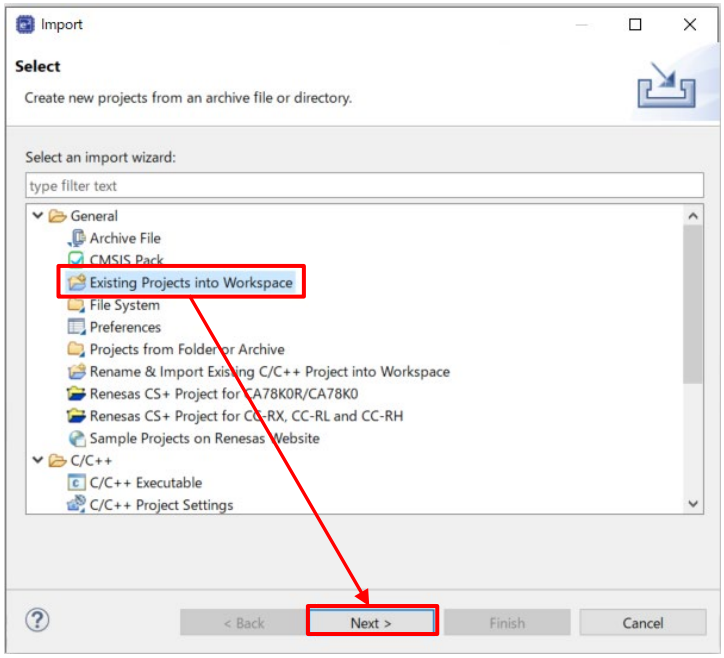
6.1 Import Modbus Sample Project

This section describes the procedure for importing the Modbus sample project and changing the evaluation board.

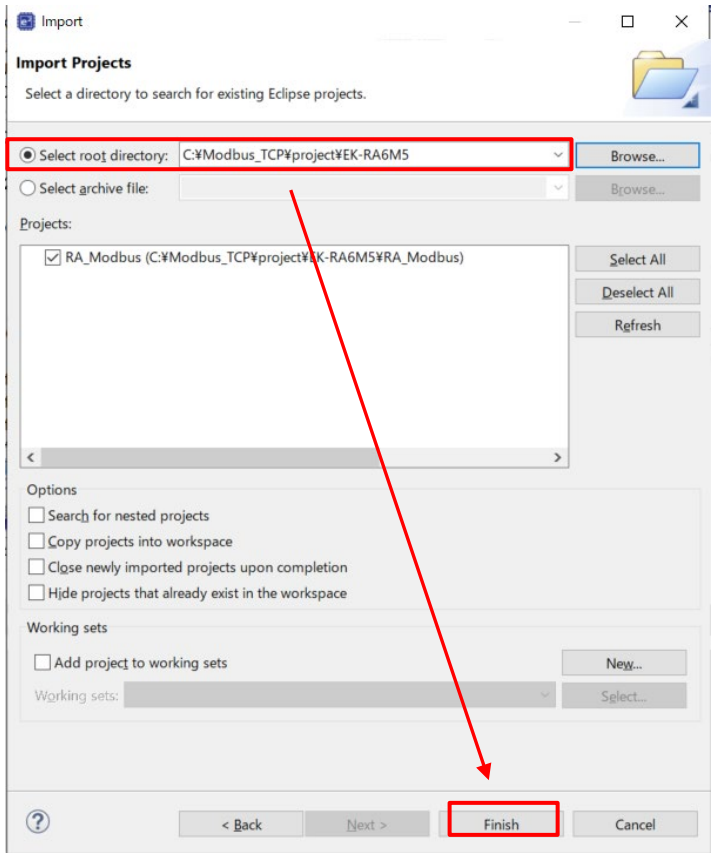
6.1.1 EK-RA6Mx / EK-RA8x1 / MCK-RA8T1 Importing Procedures

This section describes the procedures for importing EK-RA6Mx (EK-RA6M3, EK-RA6M4, EK-RA6M5) / EK-RA8x1 (EK-RA8D1, EK-RA8M1) / MCK-RA8T1.

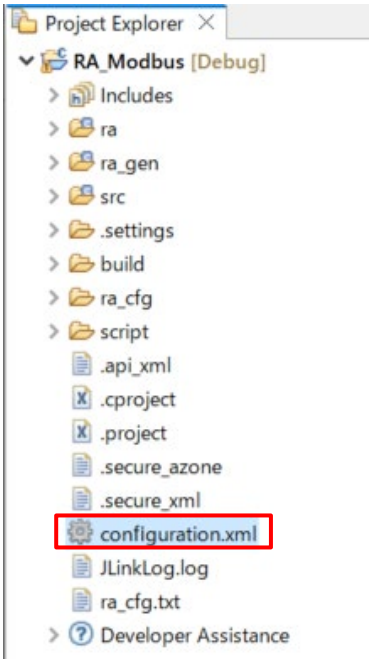
1. Import the sample project. Start e^2 studio, select [File] → [Import], and when the Import window appears, select [General] → [Existing Projects into Workspace].



Check "select root directory" and select the Modbus sample project folder (“Modbus_TCP/project/EK-RA6M5” or ”Modbus_TCP/project/MCK-RA8T2”).

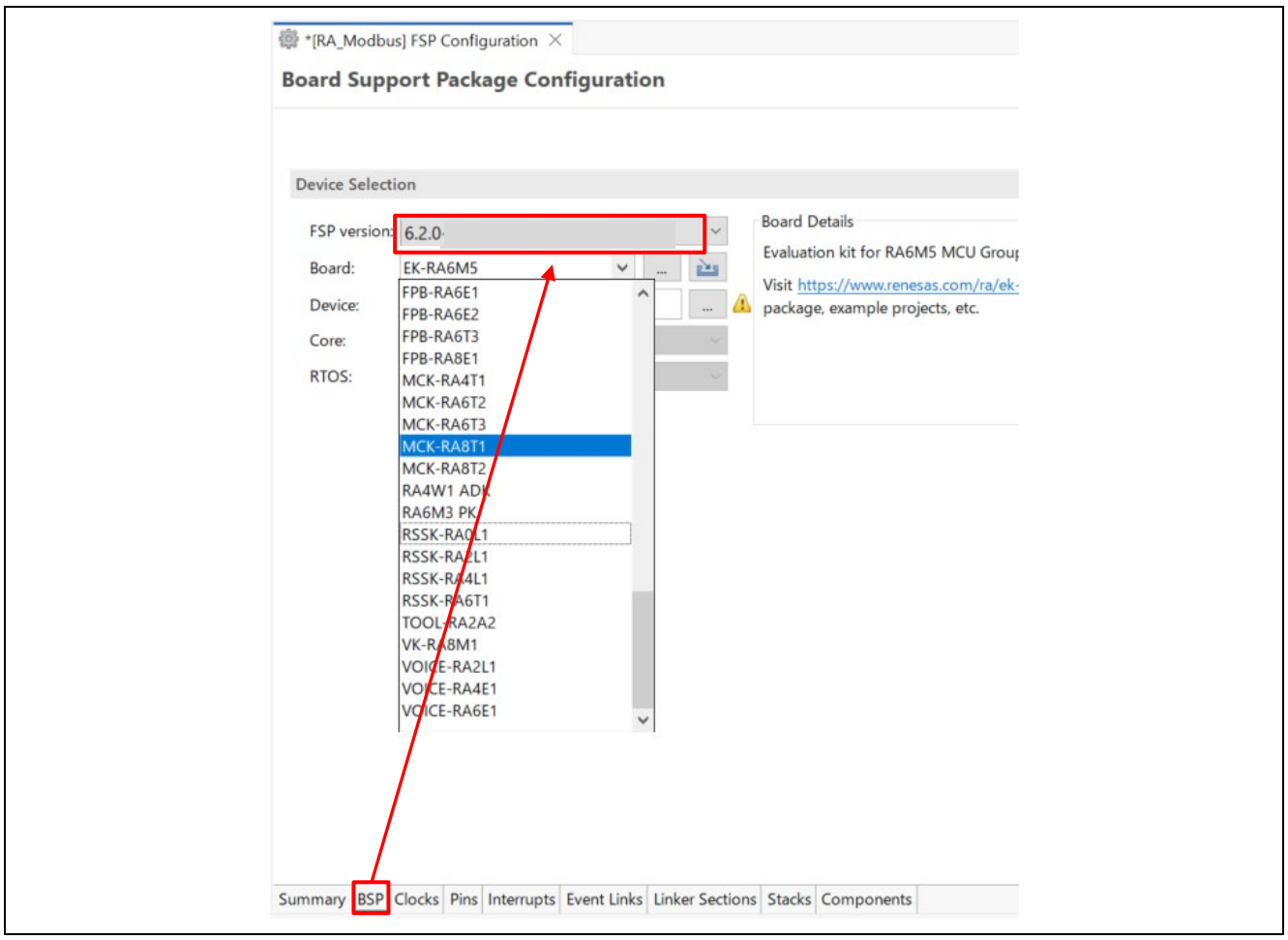


2. Open “configuration.xml” of “RA_Modbus” project.

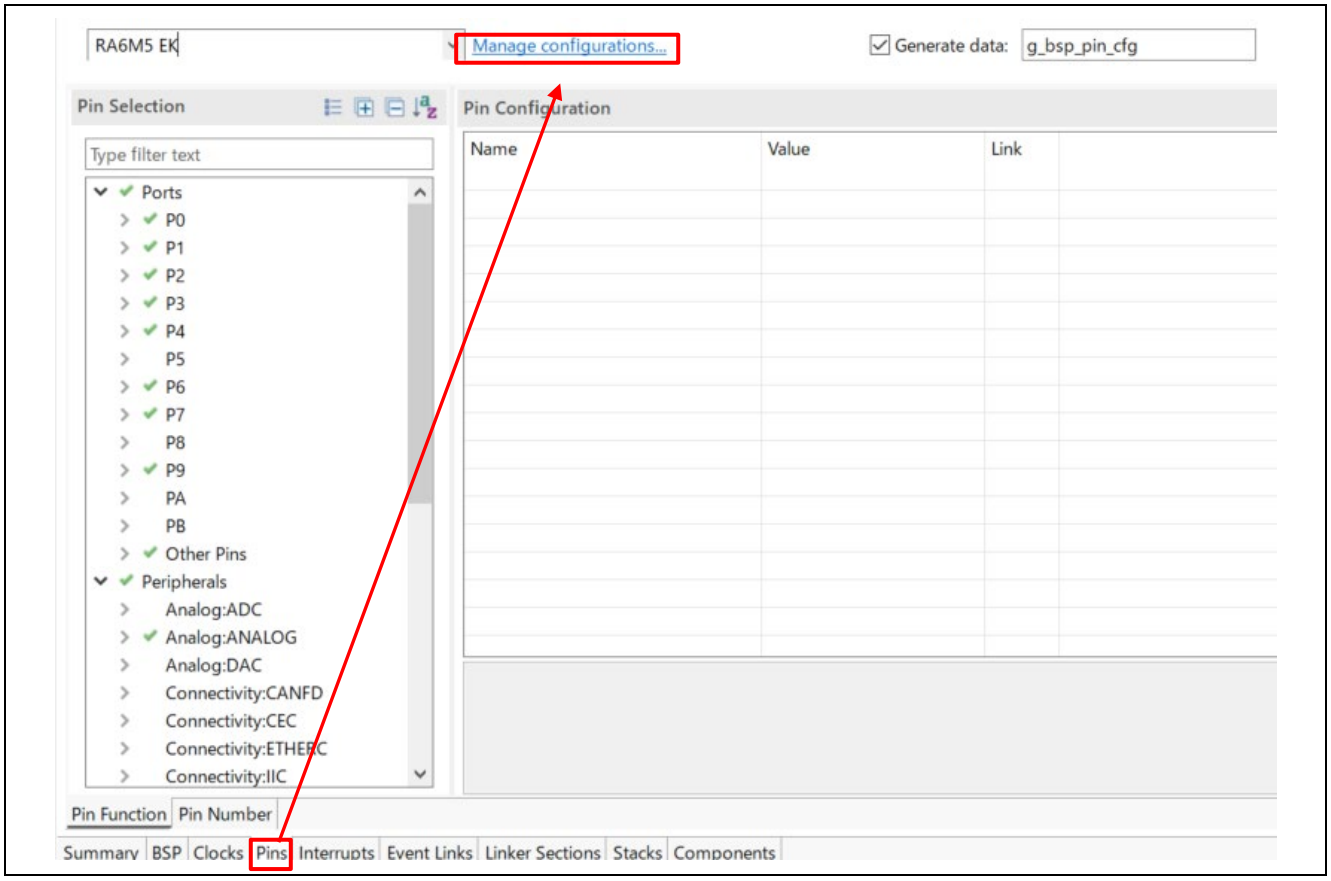


3. The "RA_Modbus" project under the "Modbus_TCP/project/EK-RA6M5" folder is created for EK-RA6M5. To run it on another supported evaluation board, modify the project by following the steps (1) to (4) below.

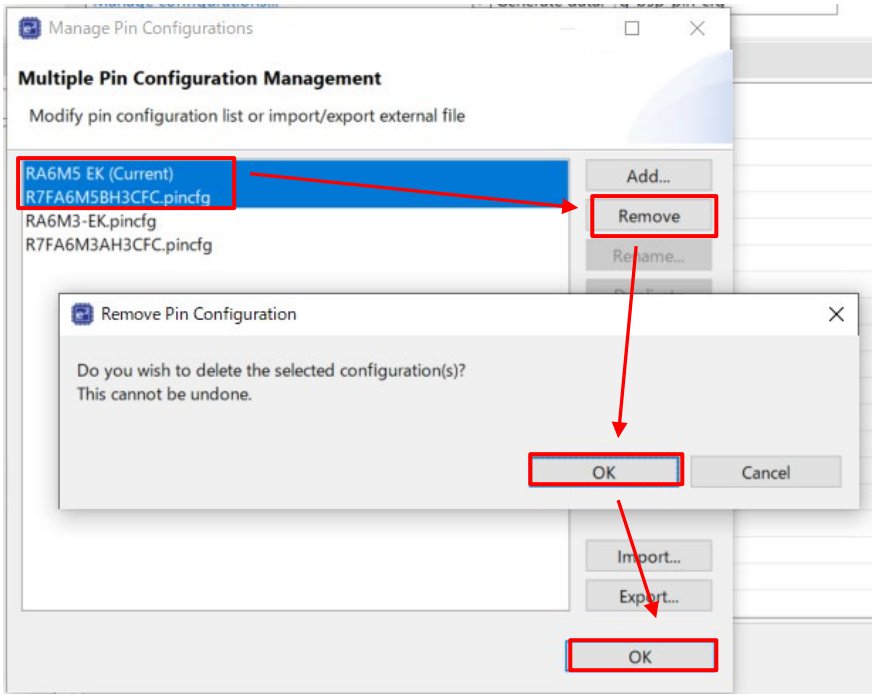
   (1) Click the "BSP" tab, then select "EK-RA6M3", "EK-RA6M4", "EK-RA8D1", "EK-RA8M1", or "MCK-RA8T1" from "Board".



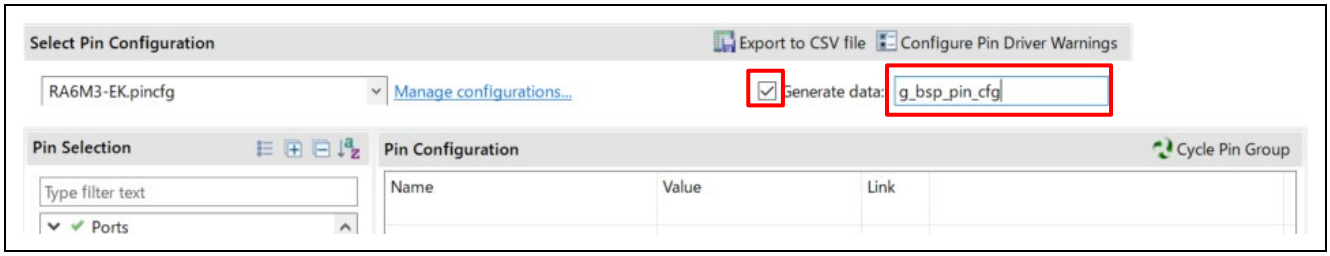
(2) Click "Manage Configurations" on the "Pins" tab



(3) Select "RA6M5 EK (Current)" and "R7FA6M5BH3CFC.pincfg" and click "Remove". Click "OK" in the "Remove Pin Configuration" pop-up, then click "OK" in "Manage Pin Configurations".

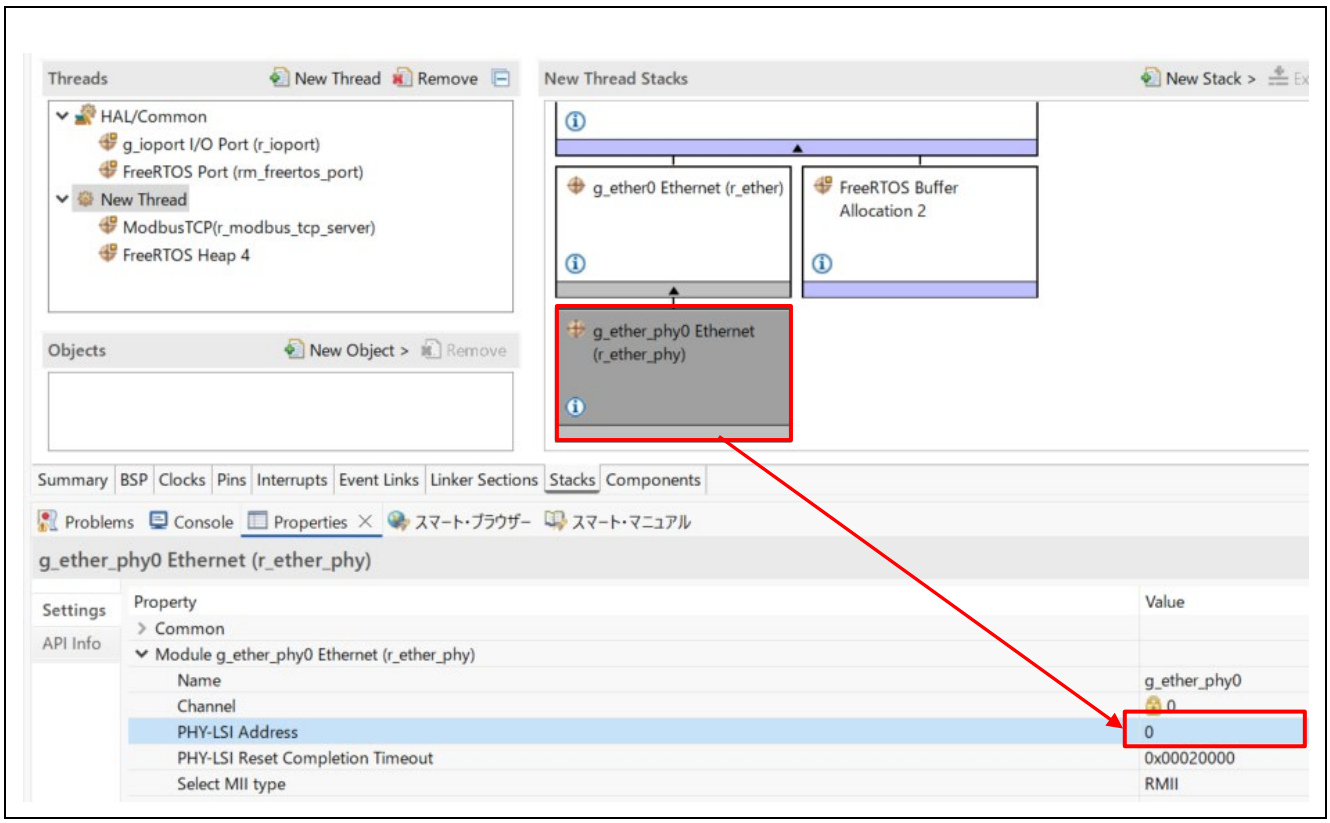


(4) Check the “Generate data” checkbox and enter “g_bsp_pin_cfg” in the text box.



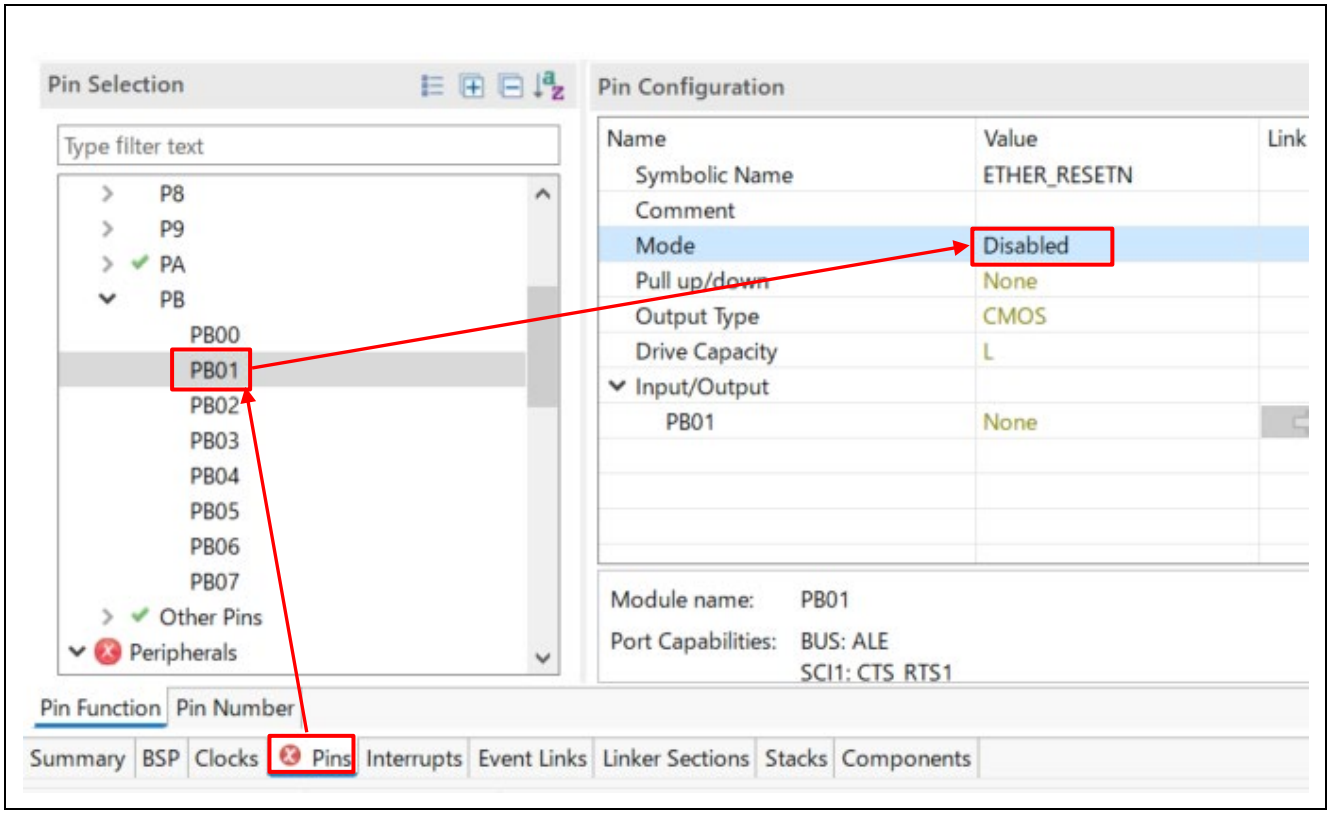
4. For "EK-RA8D1", "EK-RA8M1" and "MCK-RA8T1", change "PHY-LSI Address" in "Stacks" → "g_ether_phy0 Ethernet (r_ether_phy)" → "Module g_ether_phy0 Ethernet (r_ether_phy)" to the following value.

   PHY-LSI Address: **5**



5. For "EK-RA8D1" and "MCK-RA8T1", click the "Pins" tab, then select Reset pin [Reset Pin for each evaluation board] (see below), go to “Pin Configuration” and change the "Mode" to "Disabled".

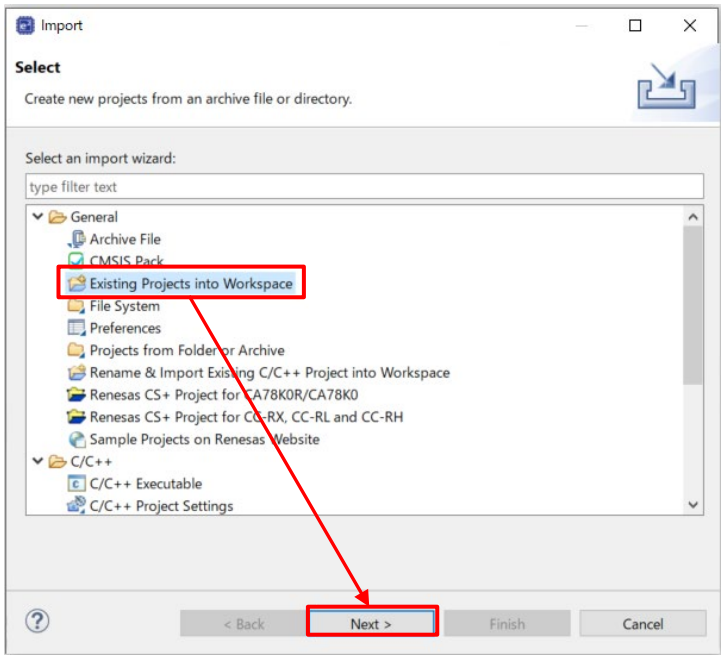
EK-RA8D1: **P706,** MCK-RA8T1: **PB01**



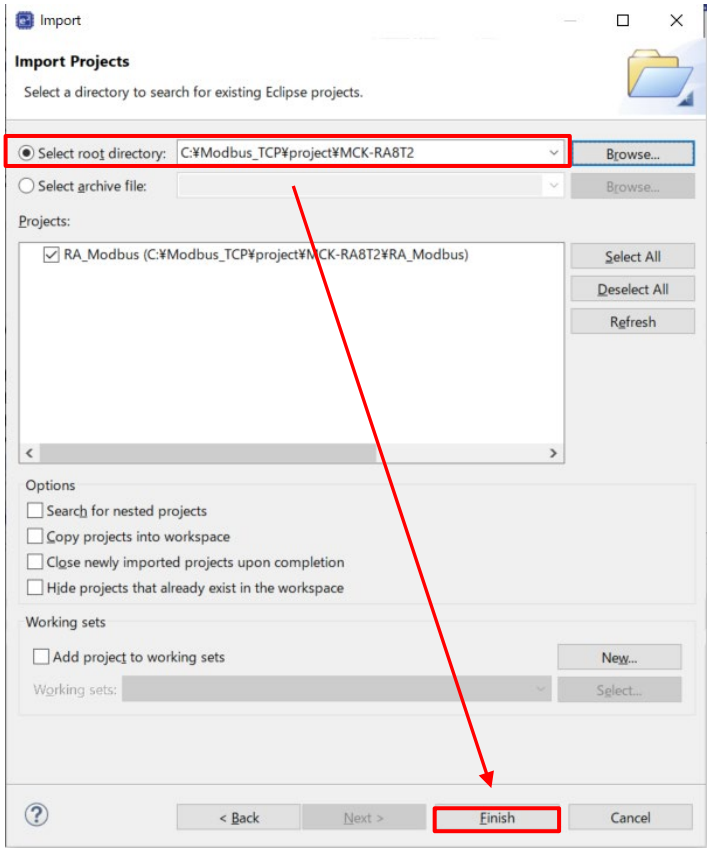
6.1.2 MCK-RA8T2 / EK-RA8D2 / EK-RA8P1 / EK-RA8M2 Importing Procedures

This section describes the procedures for importing MCK-RA8T2 / EK-RA8D2 / EK-RA8P1 / EK-RA8M2.

1. Import the sample project. Start e^2 studio, select [File] → [Import], and when the Import window appears, select [General] → [Existing Projects into Workspace].



Check "select root directory" and select the Modbus sample project folder (“Modbus_TCP/project/[evaluation board]”).



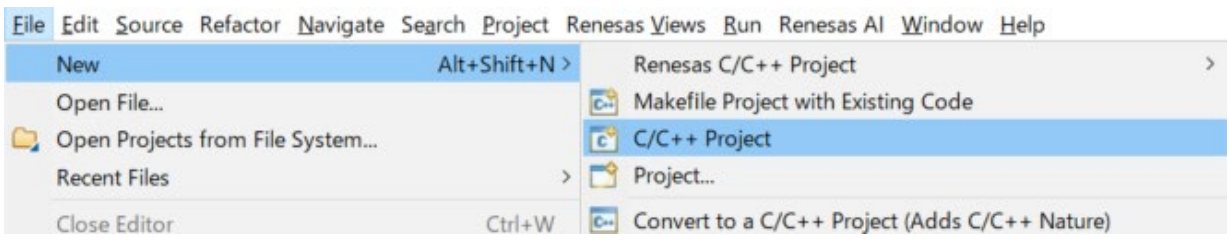
6.2 Creation of a new Modbus Project

This section describes the procedure for creating a new Modbus sample project.

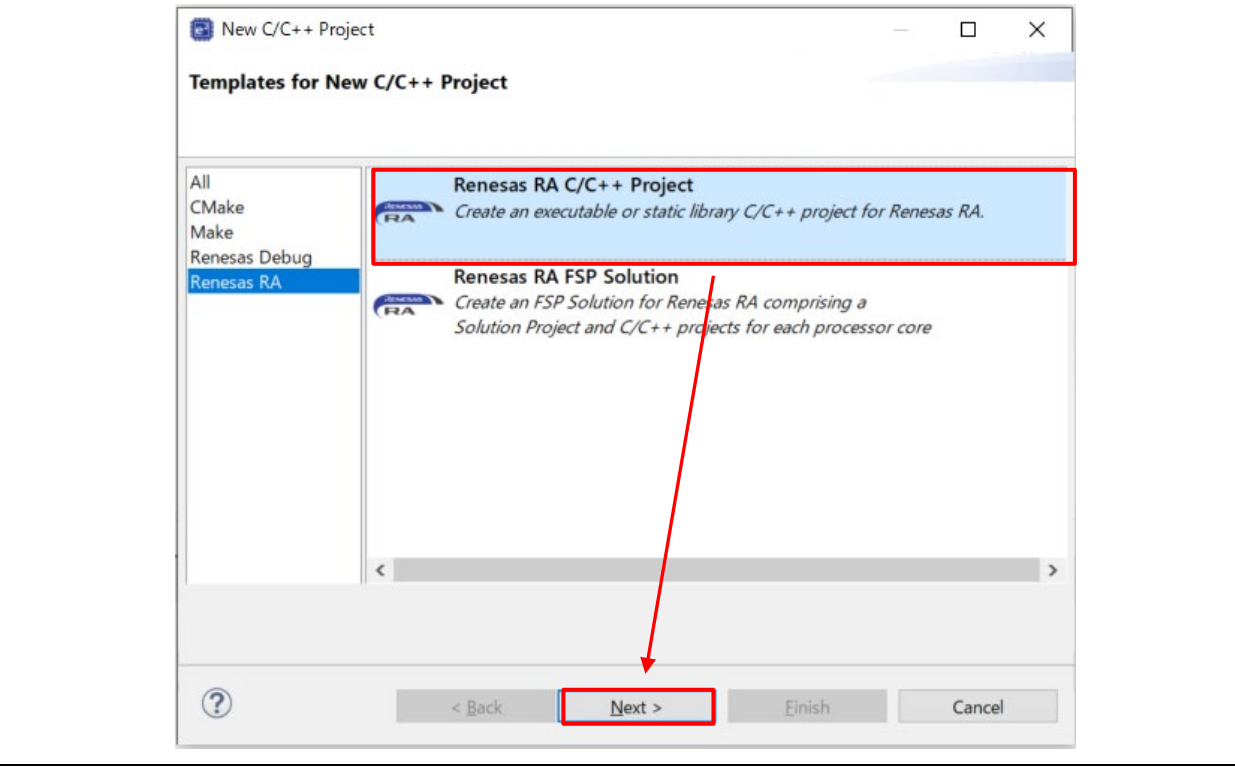
6.2.1 Common procedures for all evaluation boards

This section describes the procedures common to all evaluation boards.

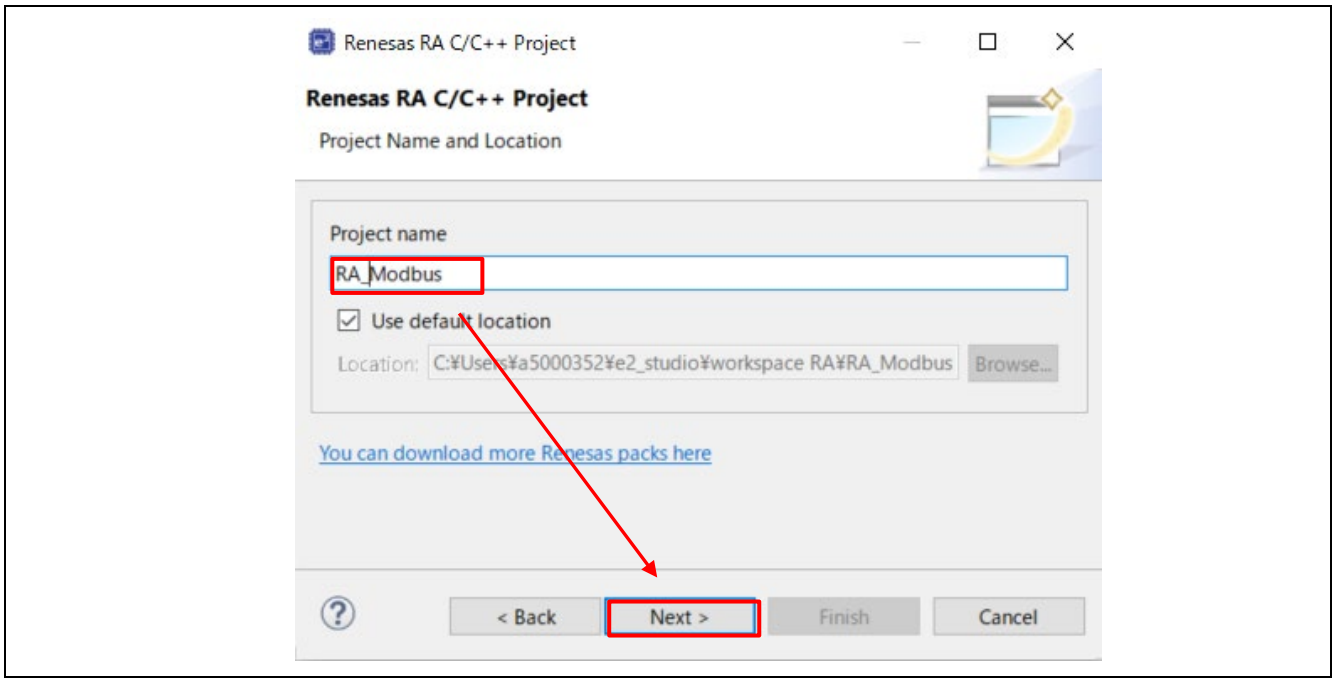
1. Set up a new project. Start e² studio and select [File] → [New] → [C/C++ Project].



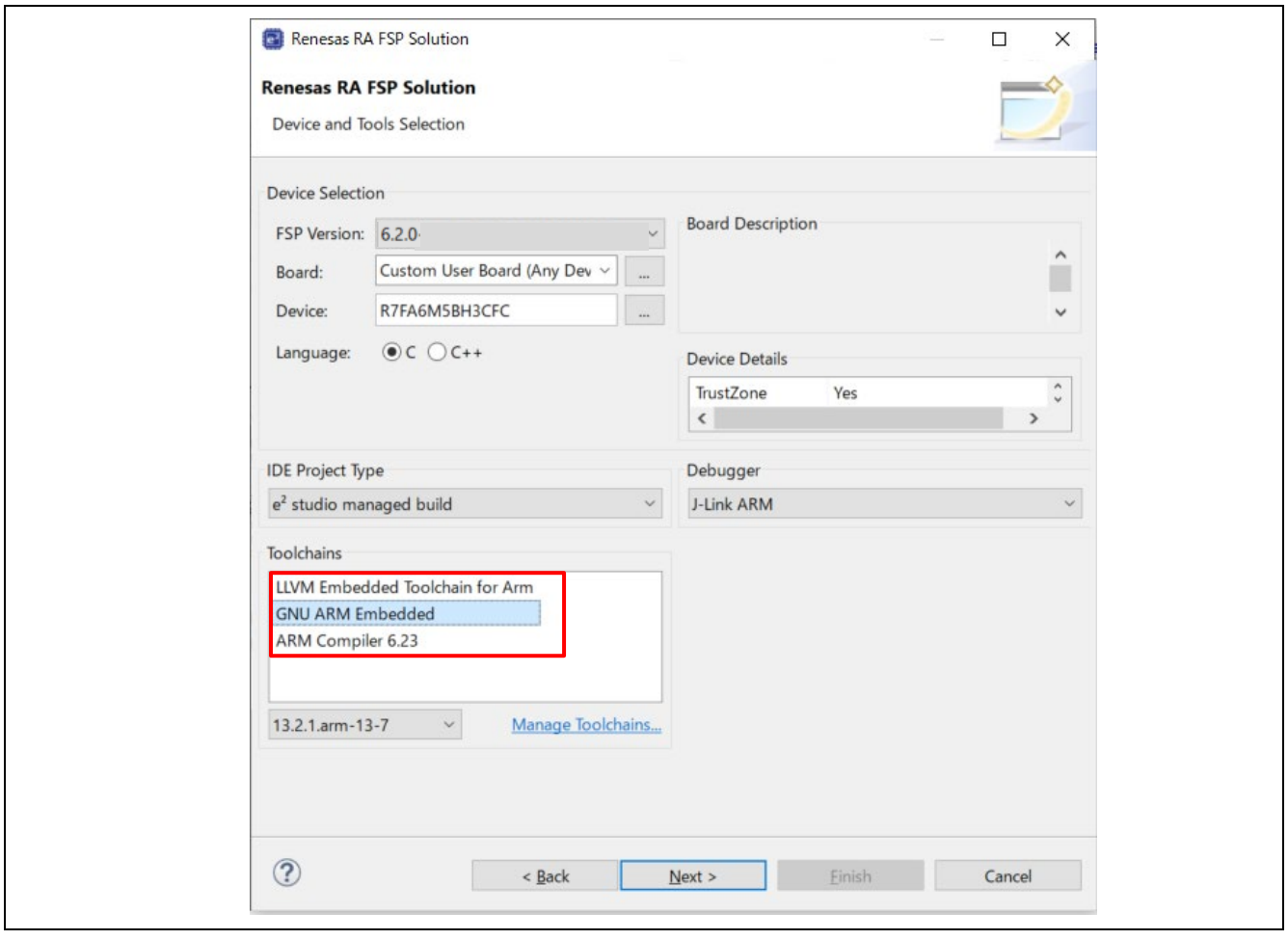
2. Select ”Renesas RA C/C++ Project”



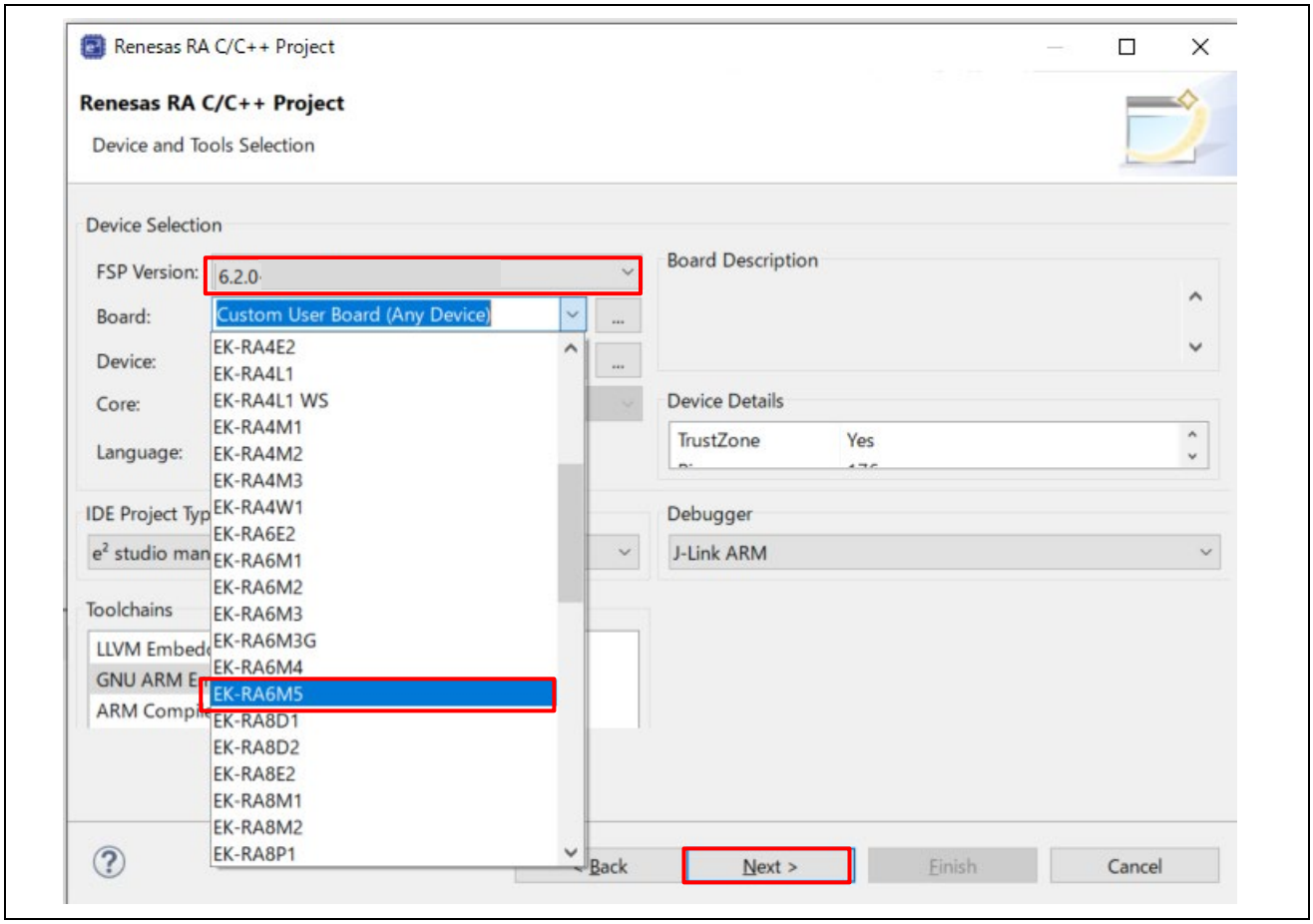
3. Enter any project name.



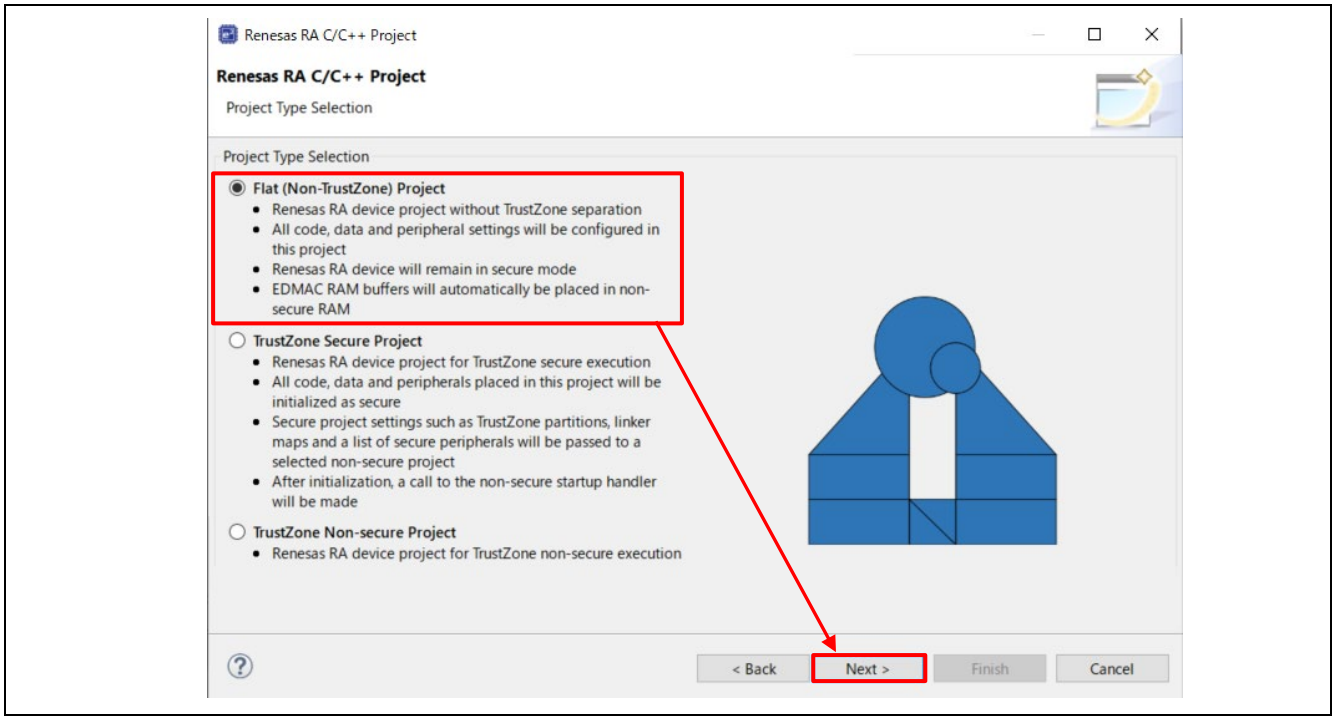
4. Select "Toolchains”



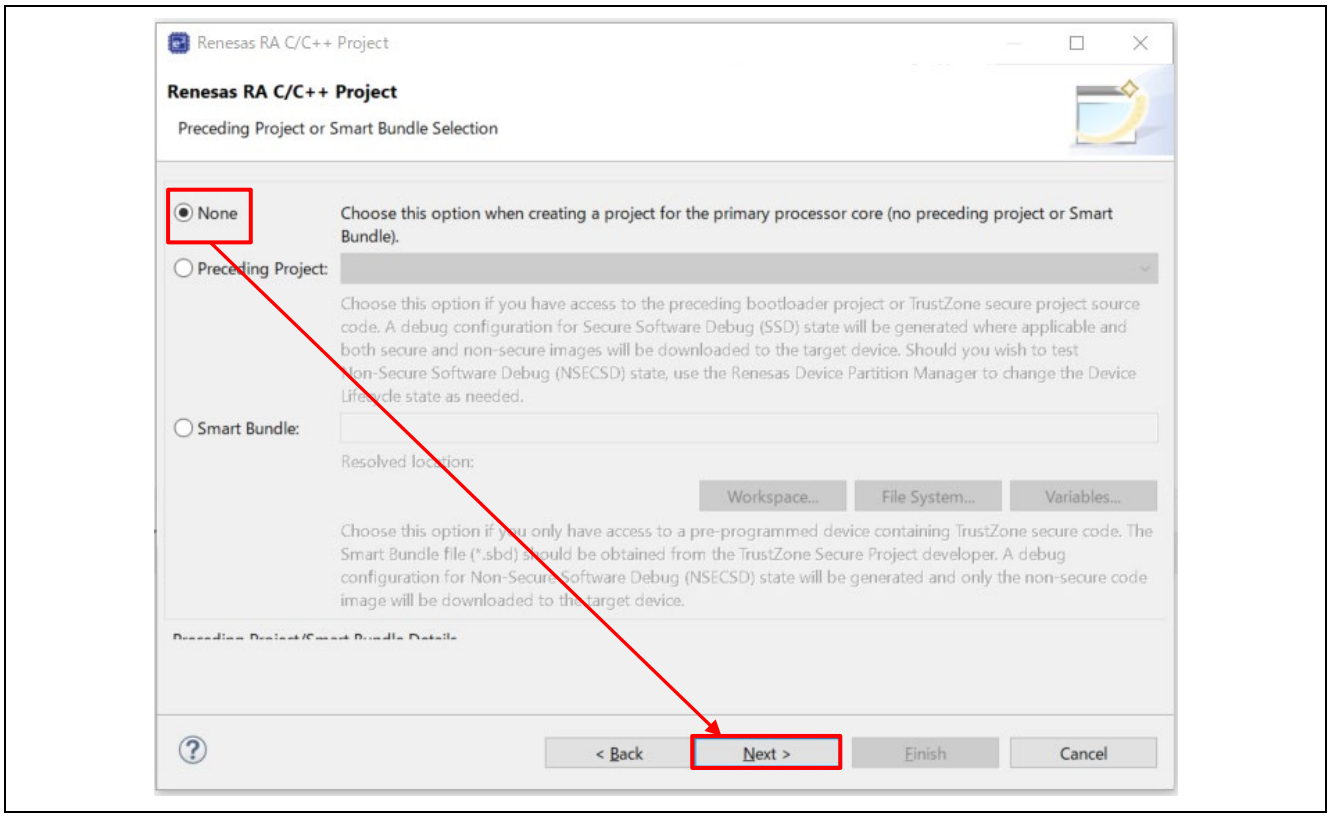
5. Select the FSP Version and Board



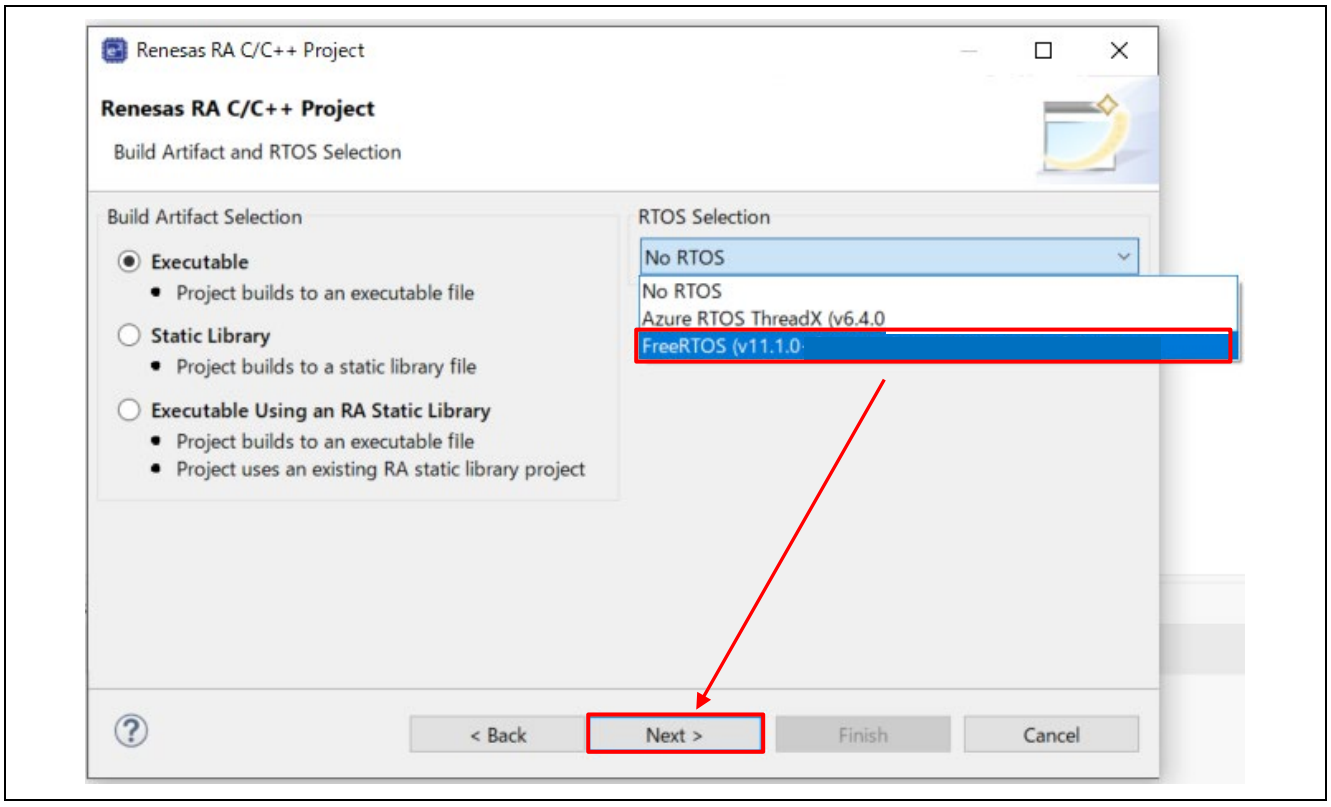
6. Select the project type for “Flat (Non-TrustZone) Project”



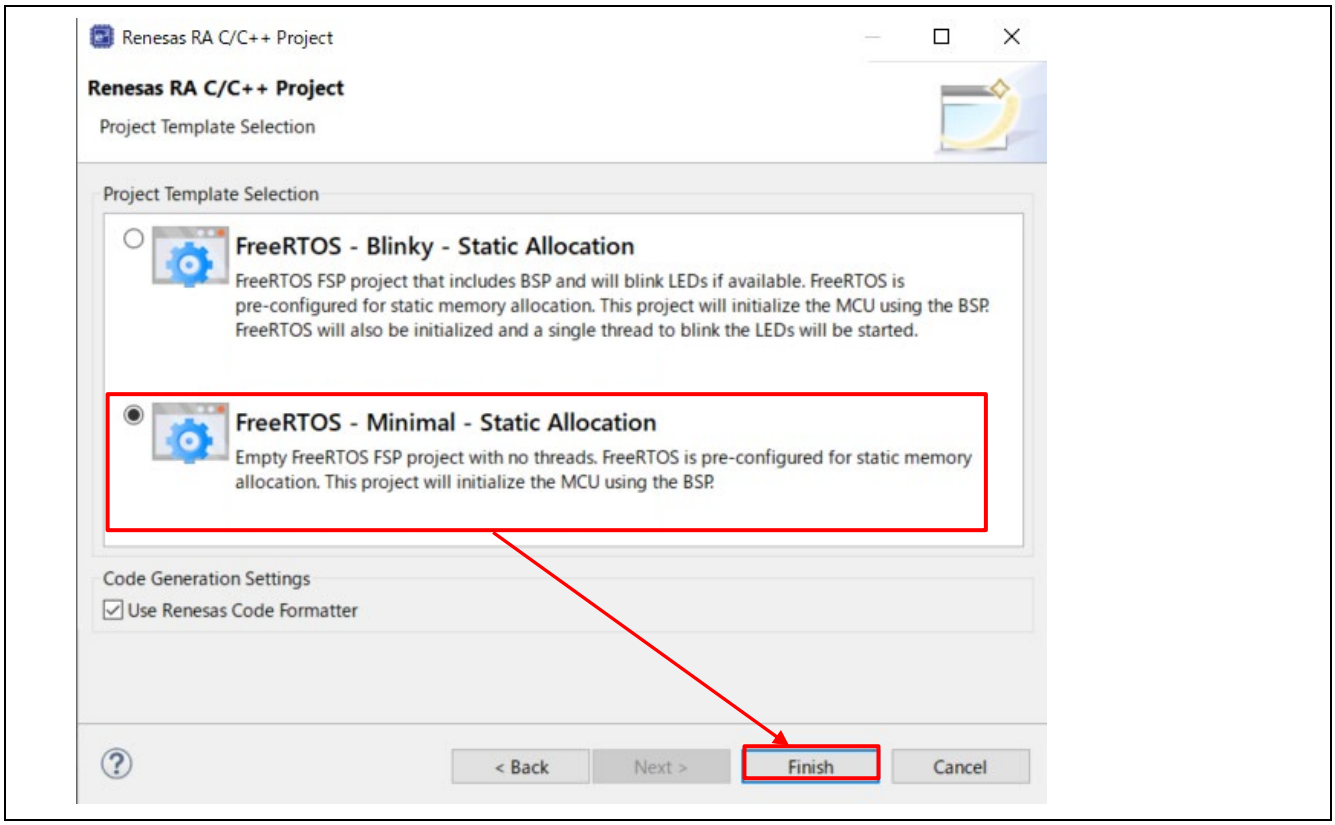
7. Select “None”



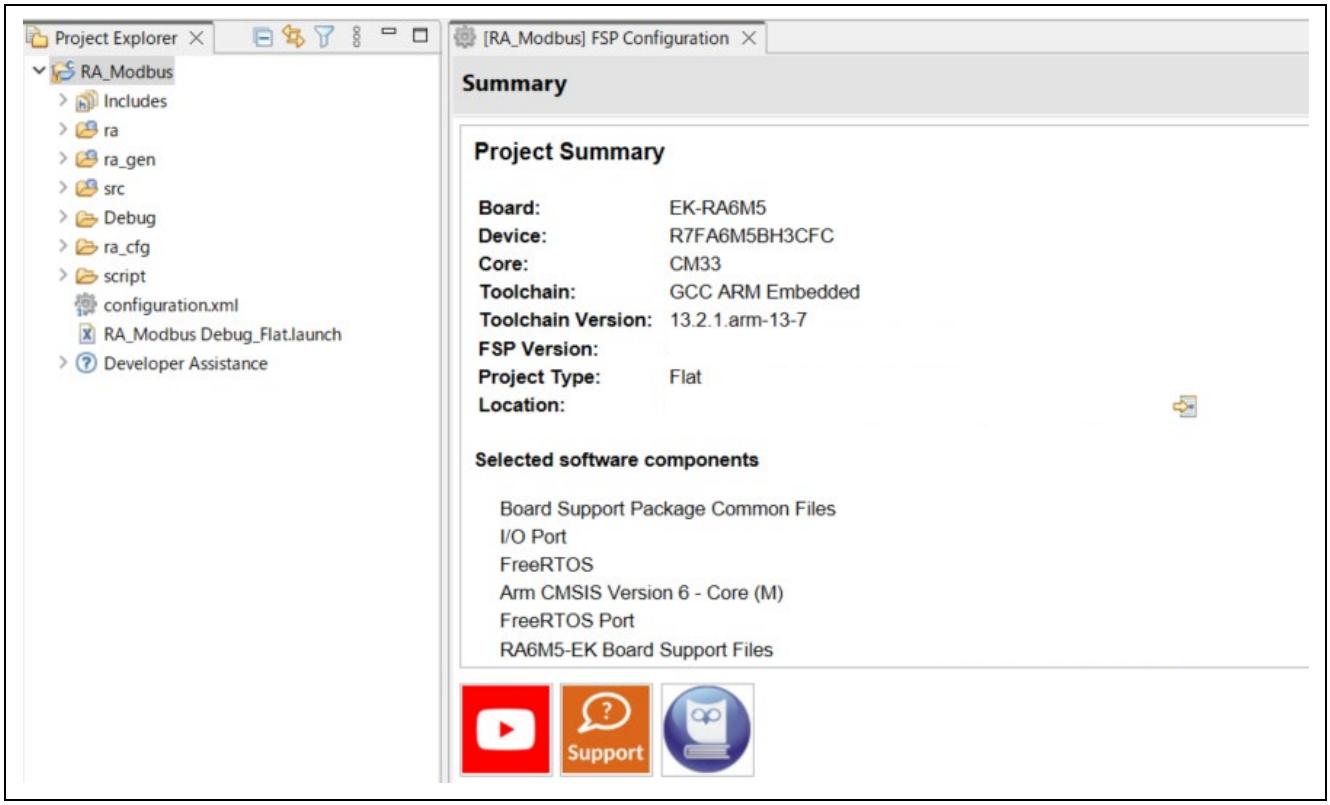
8. Select “FreeRTOS (xxx)”.



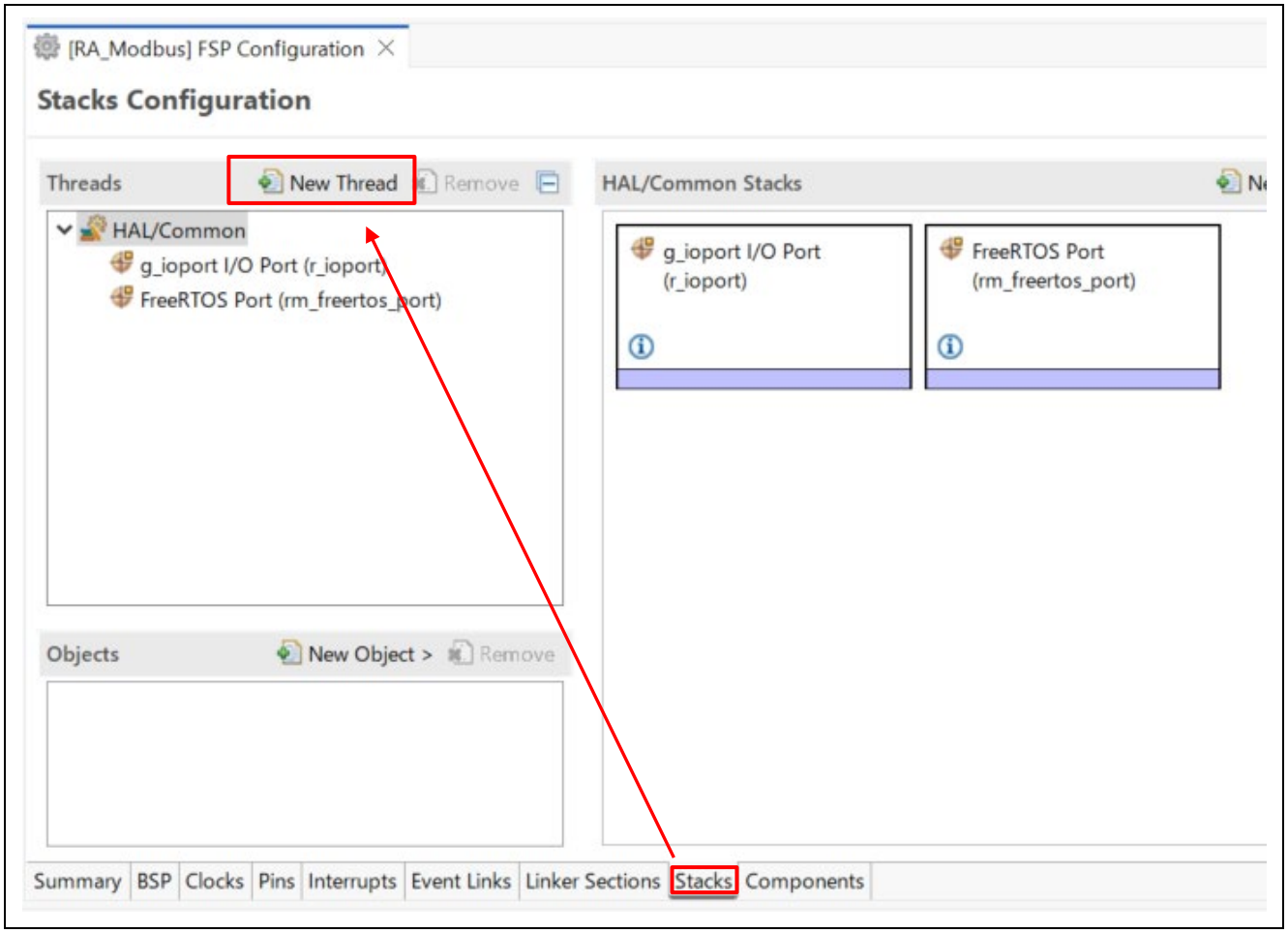
9. Select “FreeRTOS - Minimal - Static Allocation” under Project Template Selection.



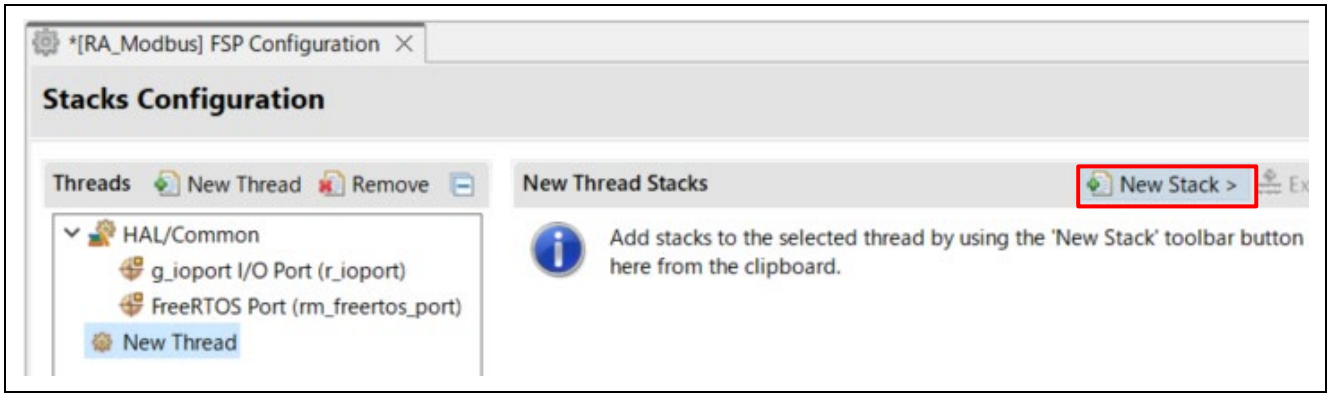
Once the project is created, it will appear in the “Project Explorer,” and the “FSP Configuration” tab will be displayed.



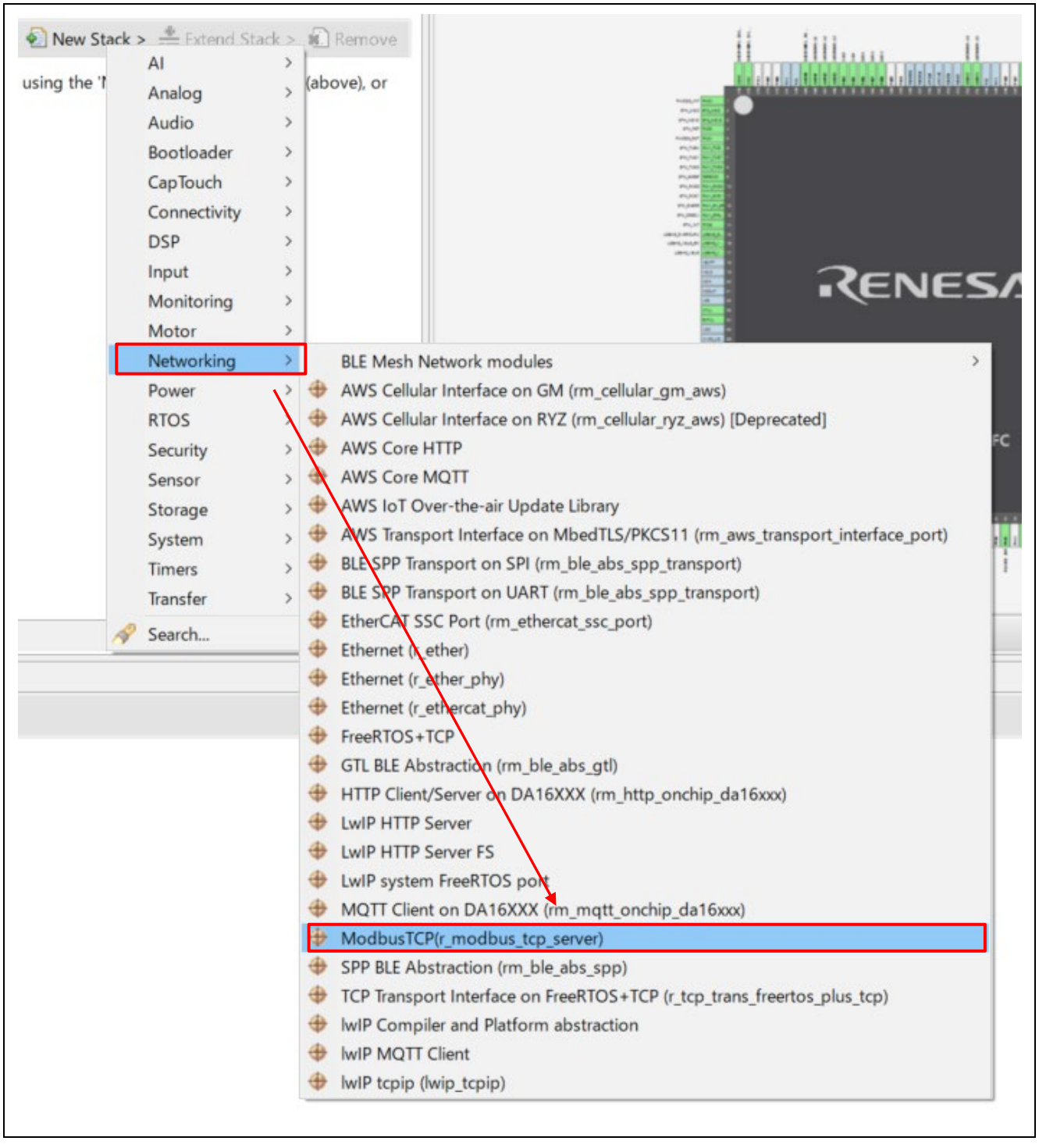
10. Create New Thread
    In the "FSP Configuration", click the "Stacks" tab, select "Threads" → "New Thread". A new thread will appear.



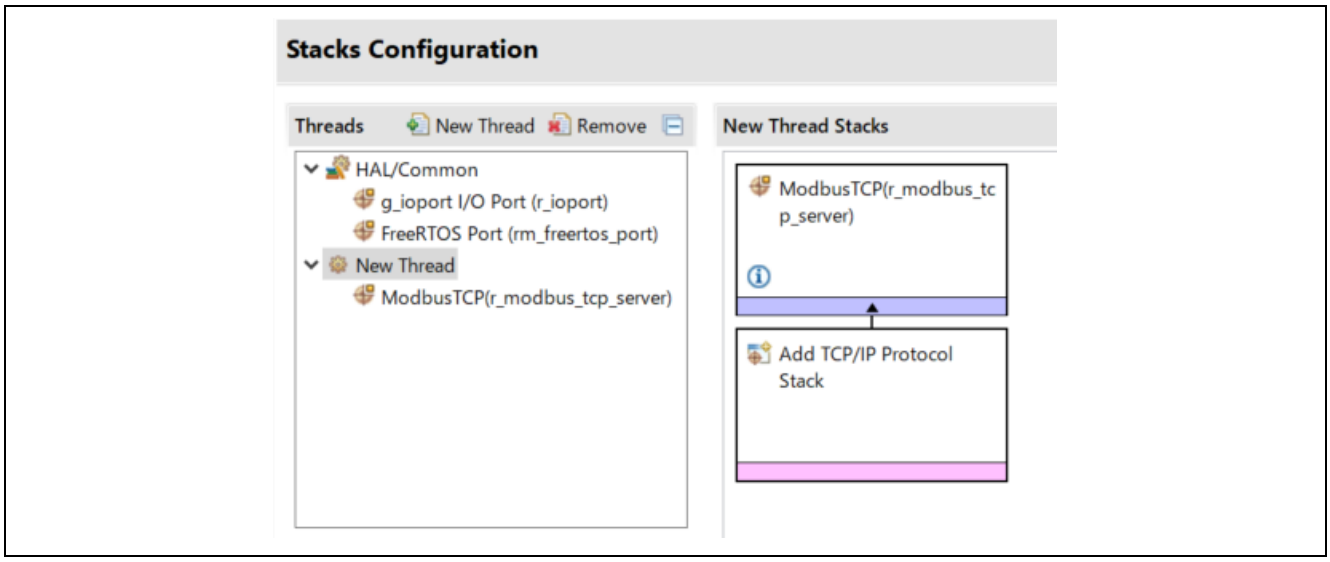
11. Create New Stack
    Select "New Stacks" to be configured.



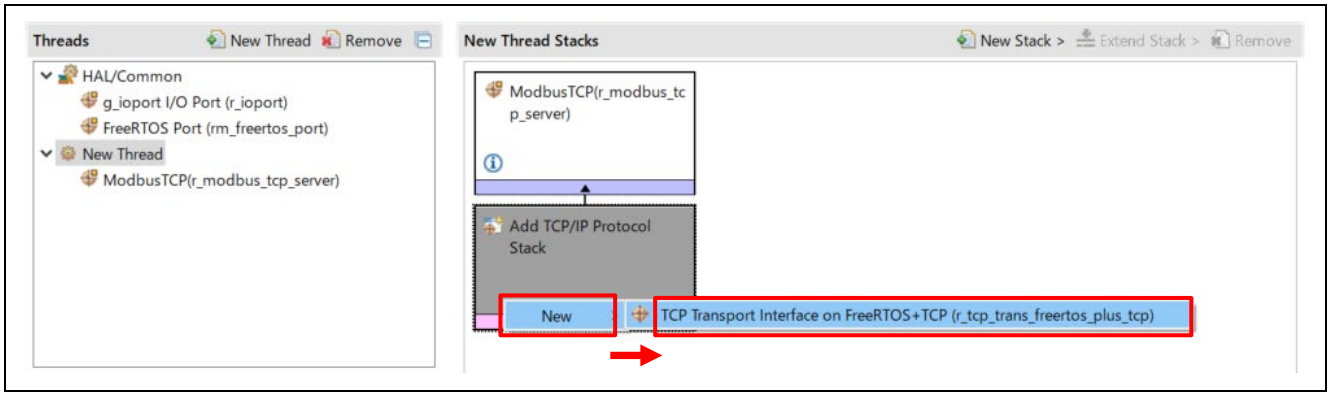
Select “Networking” → ” Modbus TCP Server (r_modbus_tcp_server)”.



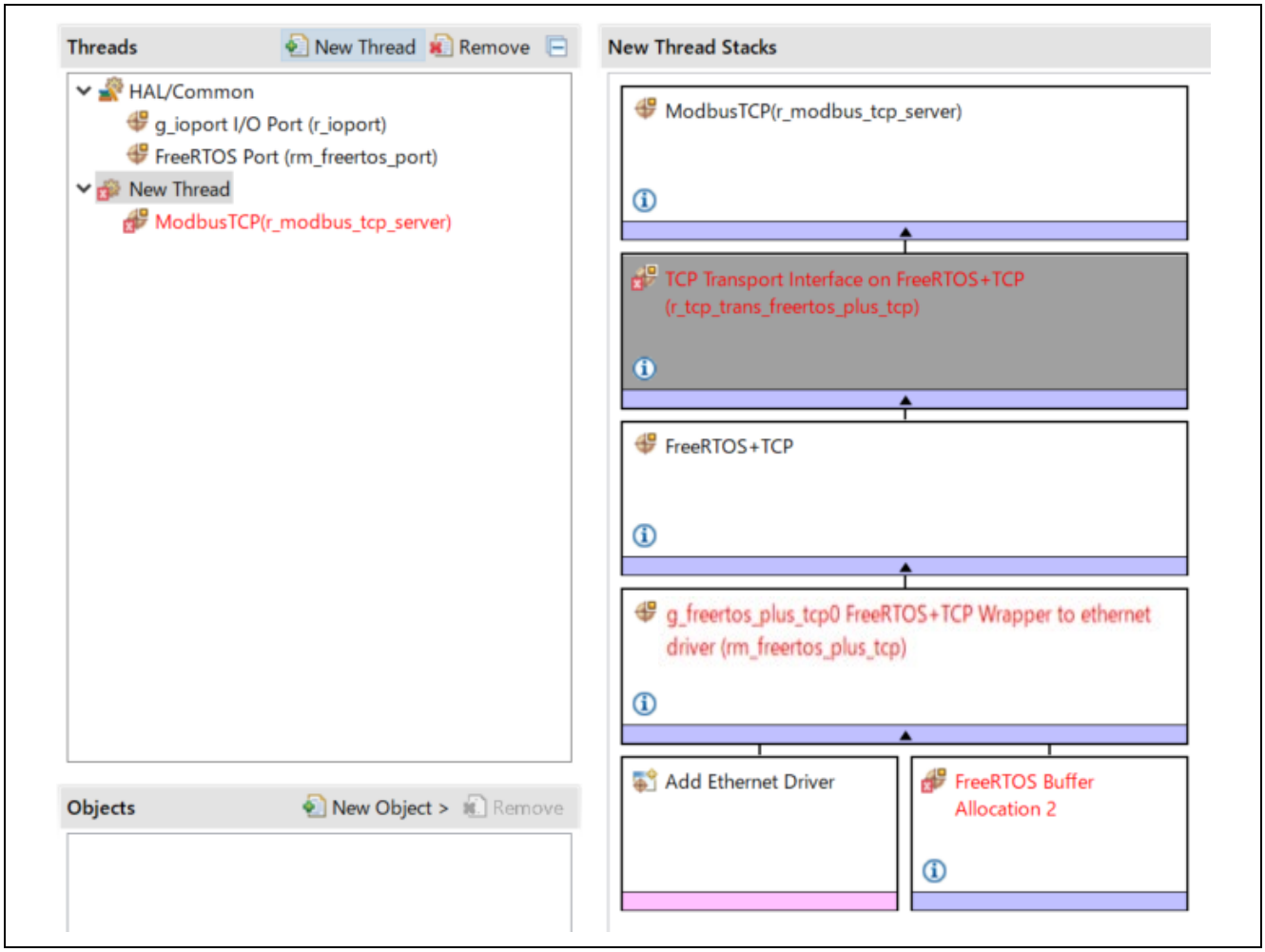
The stack is configured as follows:



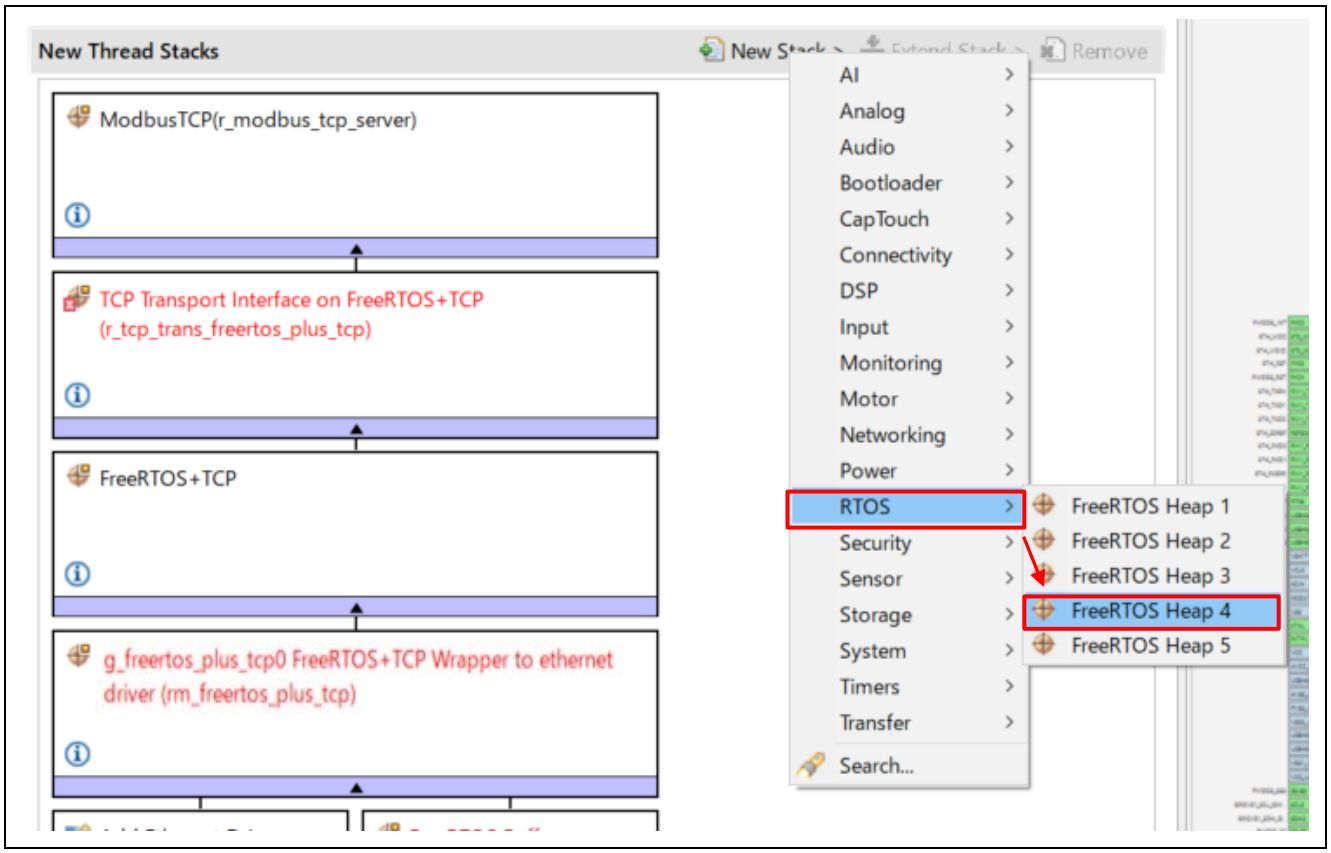
12. Add FreeRTOS+TCP
    Click “New” → “TCP Transport Interface on FreeRTOS+TCP (r_modbus_tcp_server_port_freertos_plus_tcp)” to “Add TCP/IP Protocol Stack”.



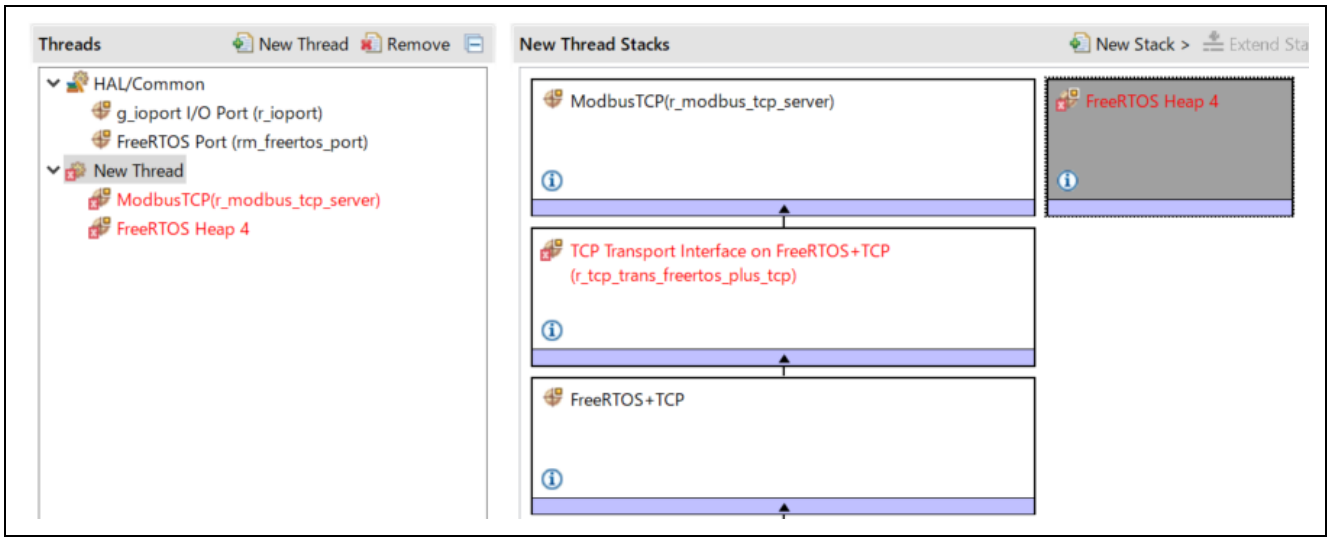
The stack is configured as follows:



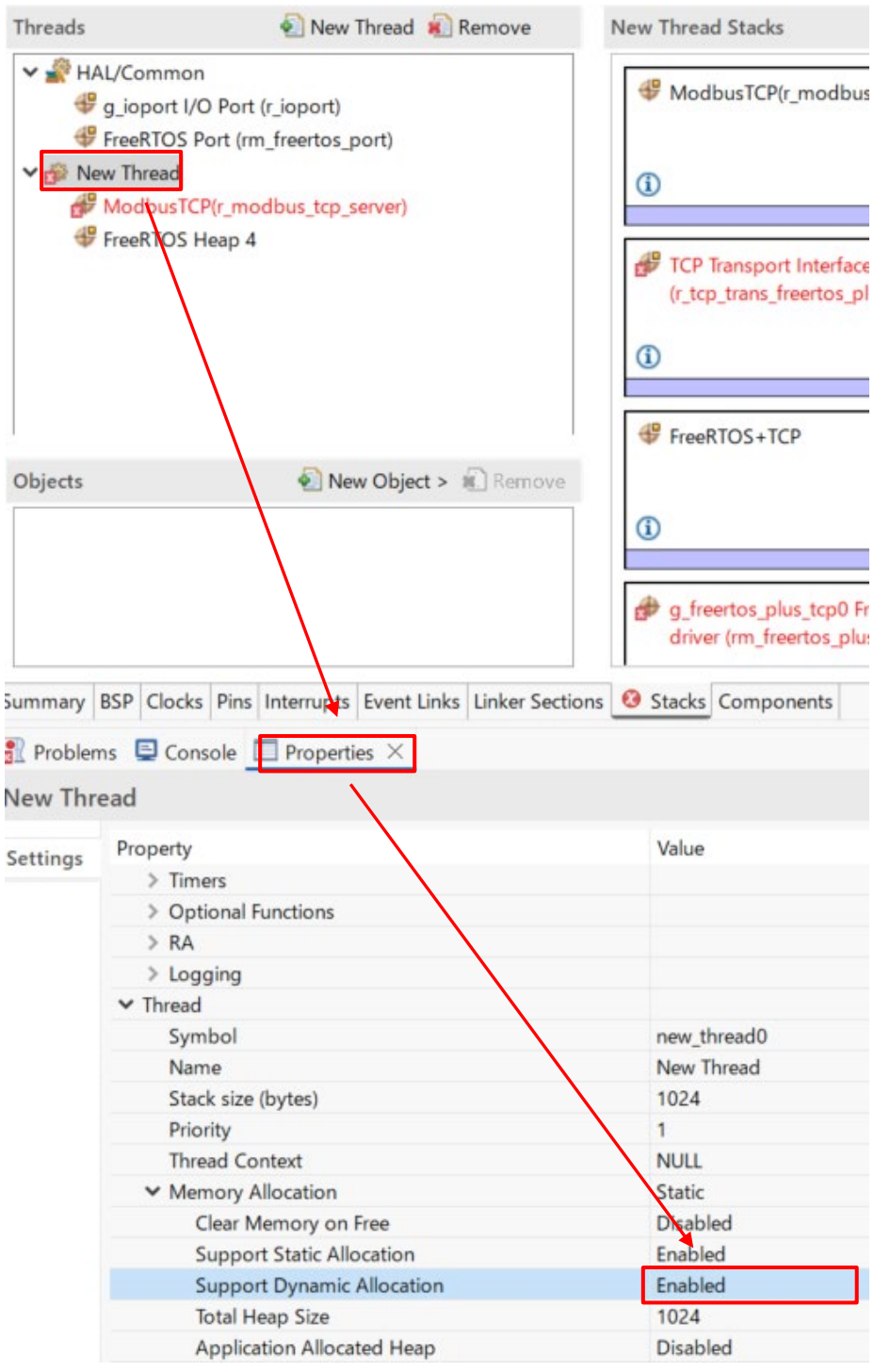
13. Add Heap
    Select “New Stack” → “RTOS” → “FreeRTOS Heap 4”.



The stack will be configured as follows:



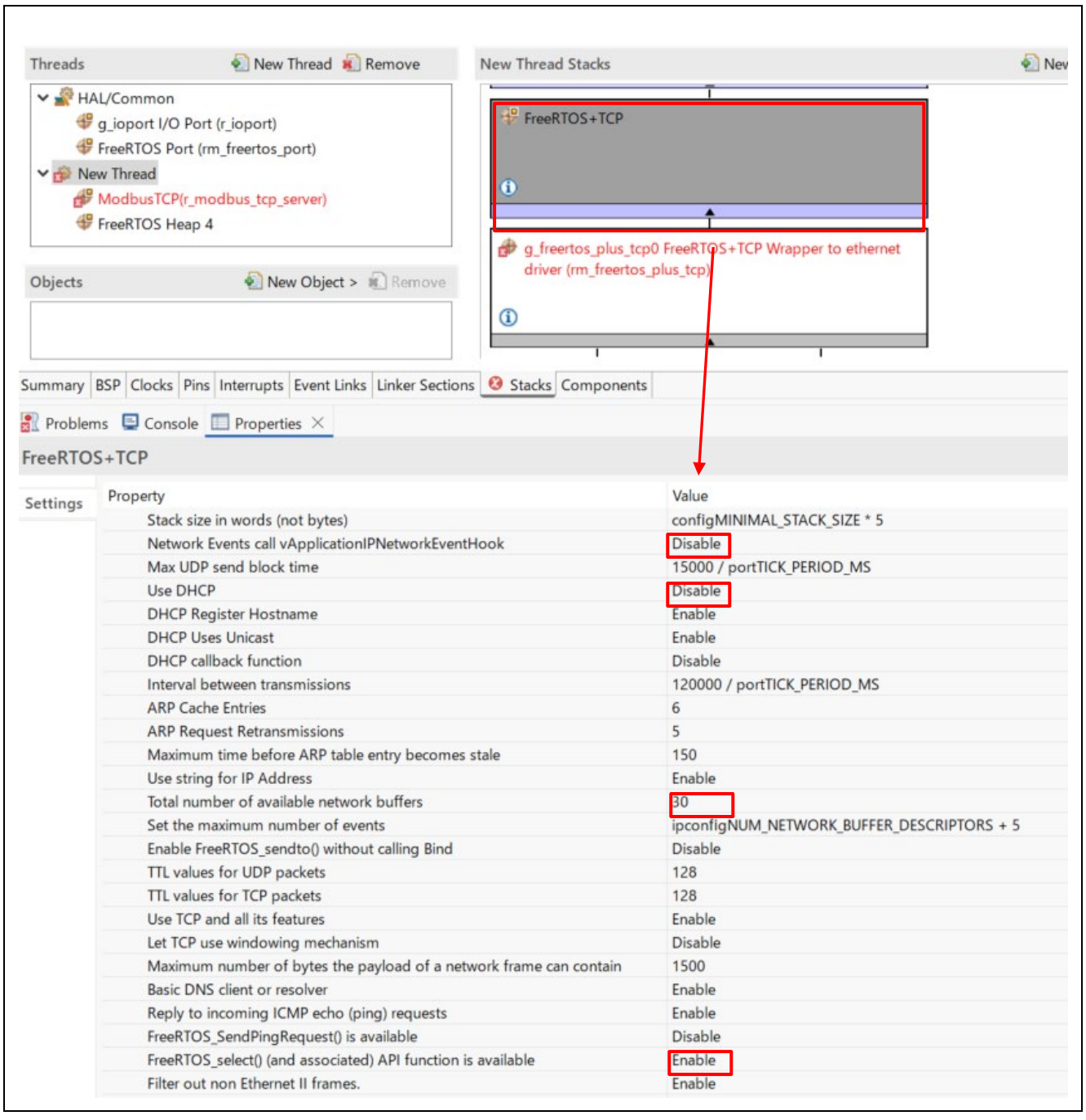
14. Set Support Dynamic Allocation
    Open “Properties” of “New Thread” in “Stacks” and change “Support Dynamic Allocation” inside “Memory Allocation” to “Enabled”.



“FreeRTOS Heap4” stack configuration errors are resolved.

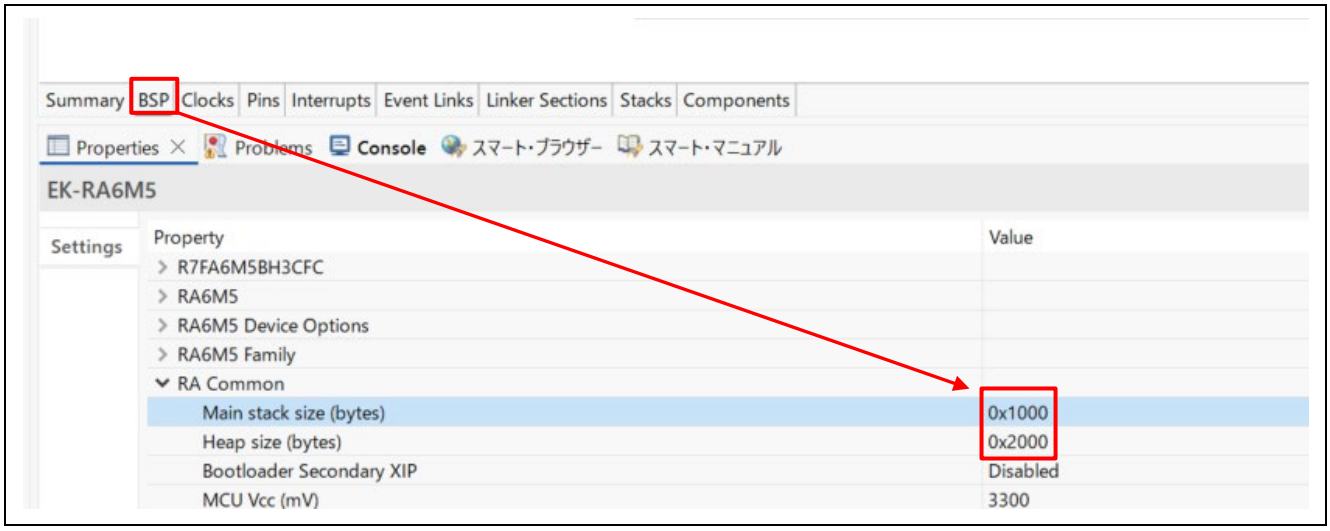
15. Set FreeRTOS+TCP
Open “Properties” of “FreeRTOS+TCP” in “Stacks” and change “Network Events call vApplicationIPNetworkEventHook”, “Use DHCP”, Total number of available network buffers and “FreeRTOS_Select() (and associated) API function is available” in “Common” to the following values.

Network Events call vApplicationIPNetworkEventHook： **Disable**
Use DHCP： **Disable**
Total number of available network buffers： **30**
FreeRTOS_Select() (and associated) API function is available： **Enable**



16. Set Stack size
    Open “Properties” of “BSP” and change “Main stack size” and “Heap size” in “RA Common” to the following values.

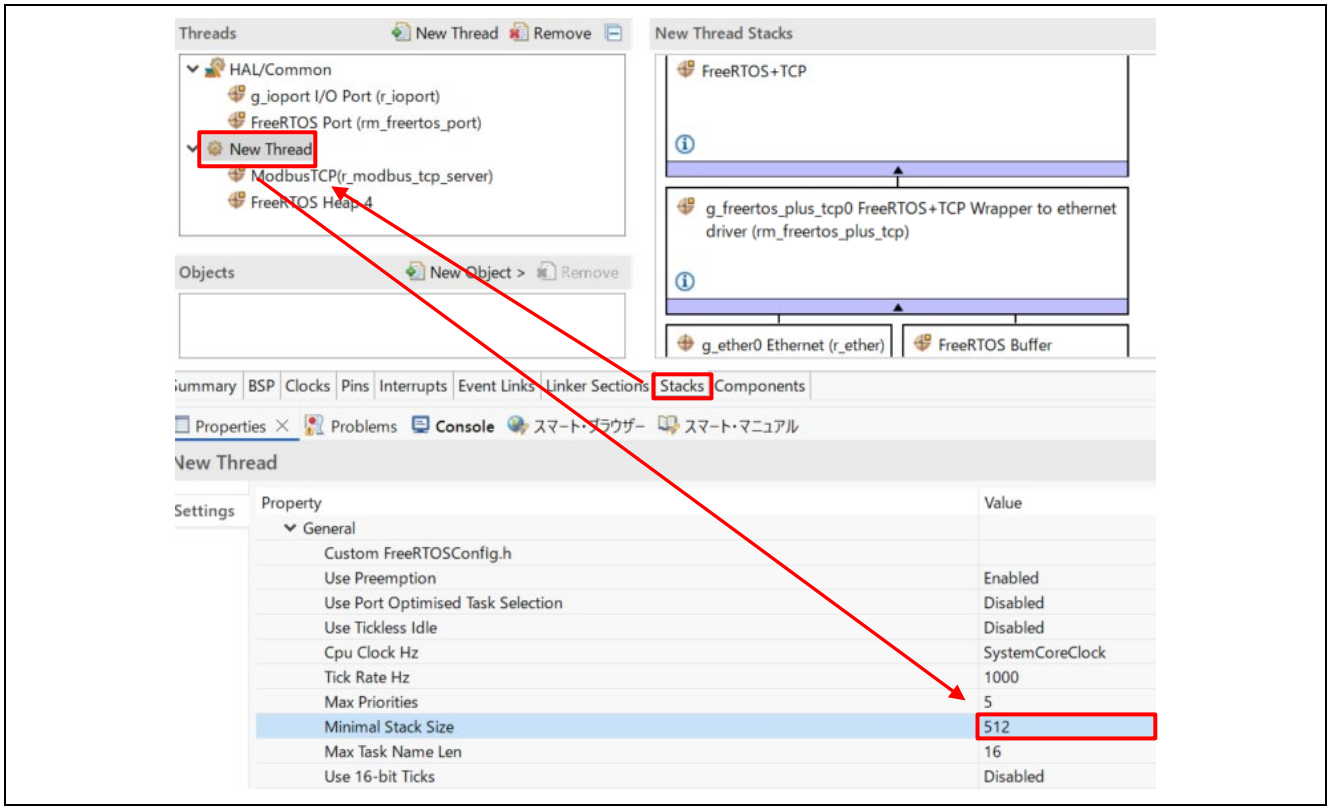
    Main stack size：**0x1000**, Heap size：**0x2000**



The stack configuration error for "ModbusTCP (r_modbus_tcp_server)" is now resolved.

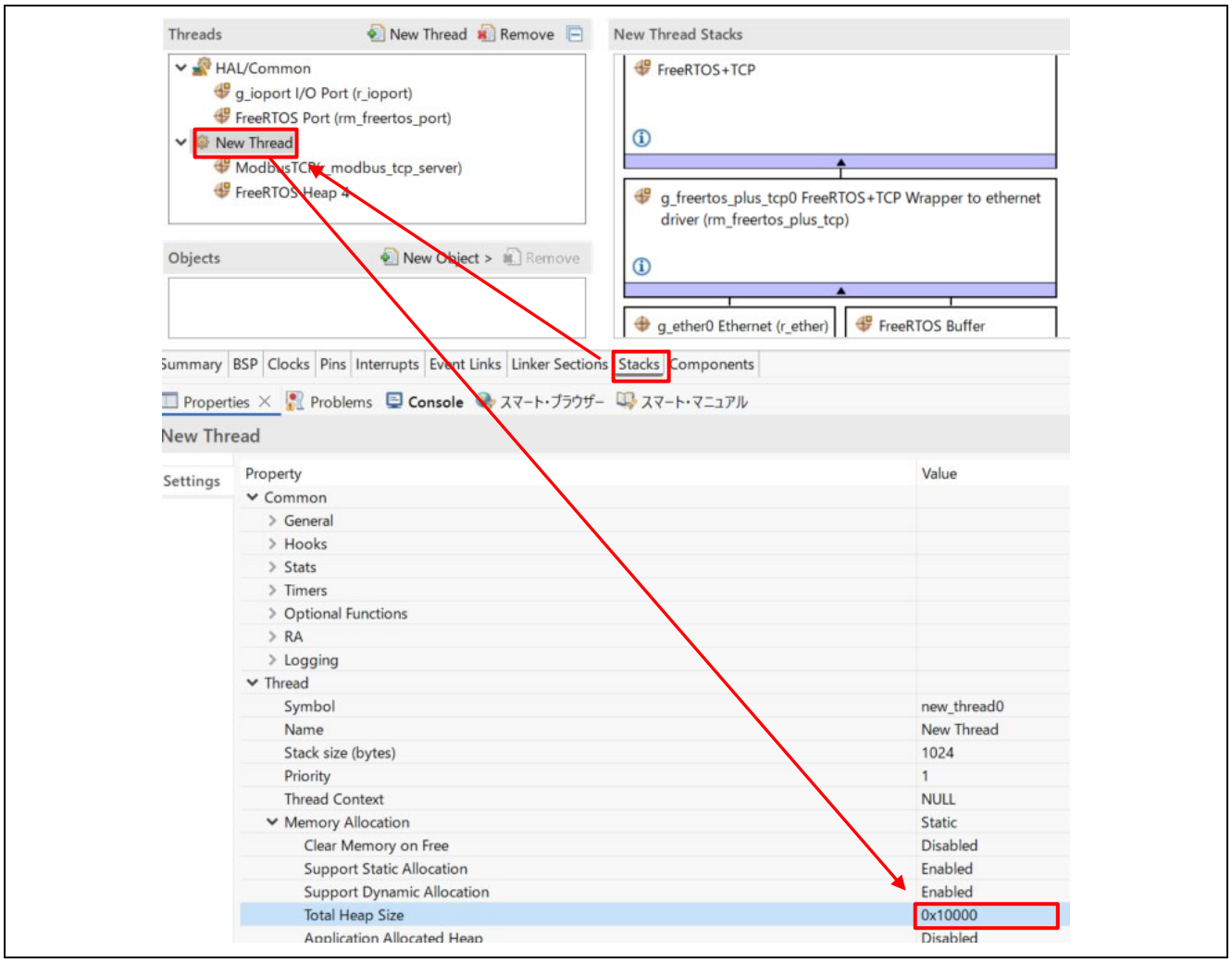
Open “Properties” of “New Thread”, then click on “Stacks” and change “Minimal Stack size” in “General” to the following values.

Minimal Stack size：**512**



Open “Properties” of “New Thread”, then click on “Stacks” and change “Total Heap Size” in “Memory Allocation” to the following values.

Total Heap Size：**0x10000**



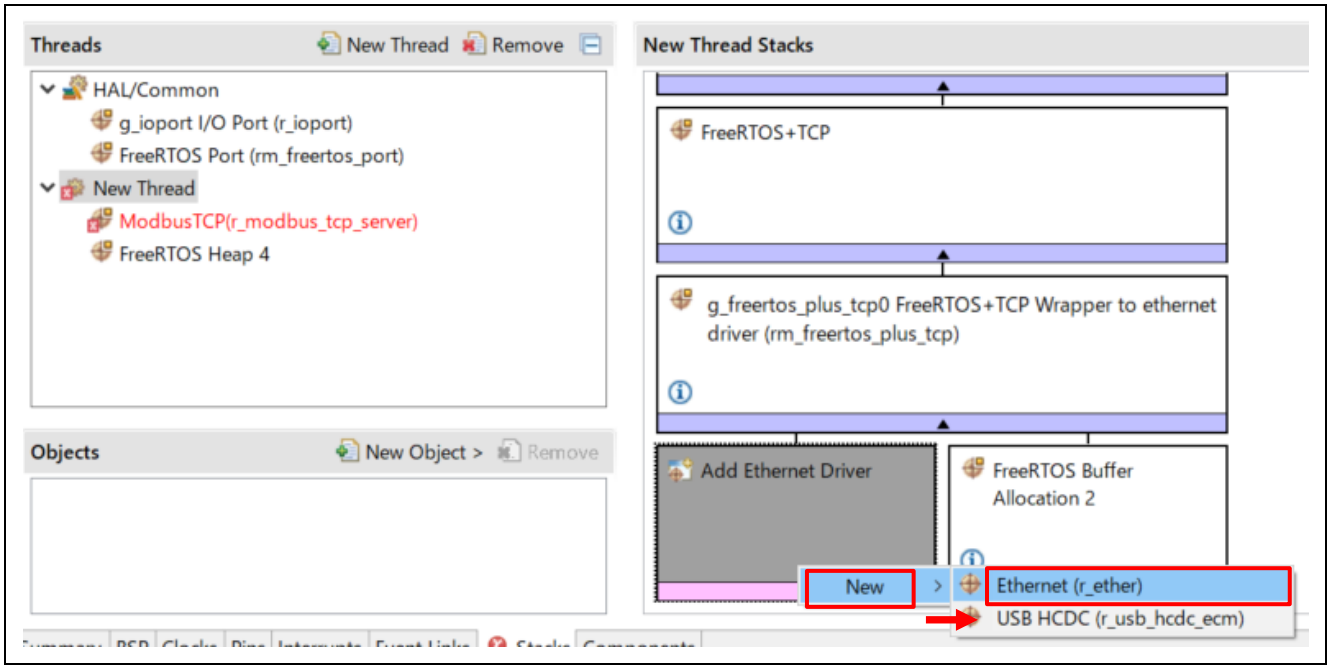
17. Perform the steps for each evaluation board.
    Refer to the following for the evaluation board you are using and follow the steps:

- EK-RA6M3, EK-RA6M4, EK-RA6M5,EK-RA8D1, EK-RA8M1, MCK-RA8T1 :
  6.2.2 EK-RA6Mx / EK-RA8x1 / MCK-RA8T1 Creating Procedures
- MCK-RA8T2 : **6.2.3. MCK-RA8T2 Creating Procedures**
- EK-RA8D2 : **6.2.4. EK-RA8D2 Creating Procedures**
- EK-RA8P1 : **6.2.5. EK-RA8P1 Creating Procedures**
- EK-RA8M2 : **6.2.6. EK-RA8M2 Creating Procedures**

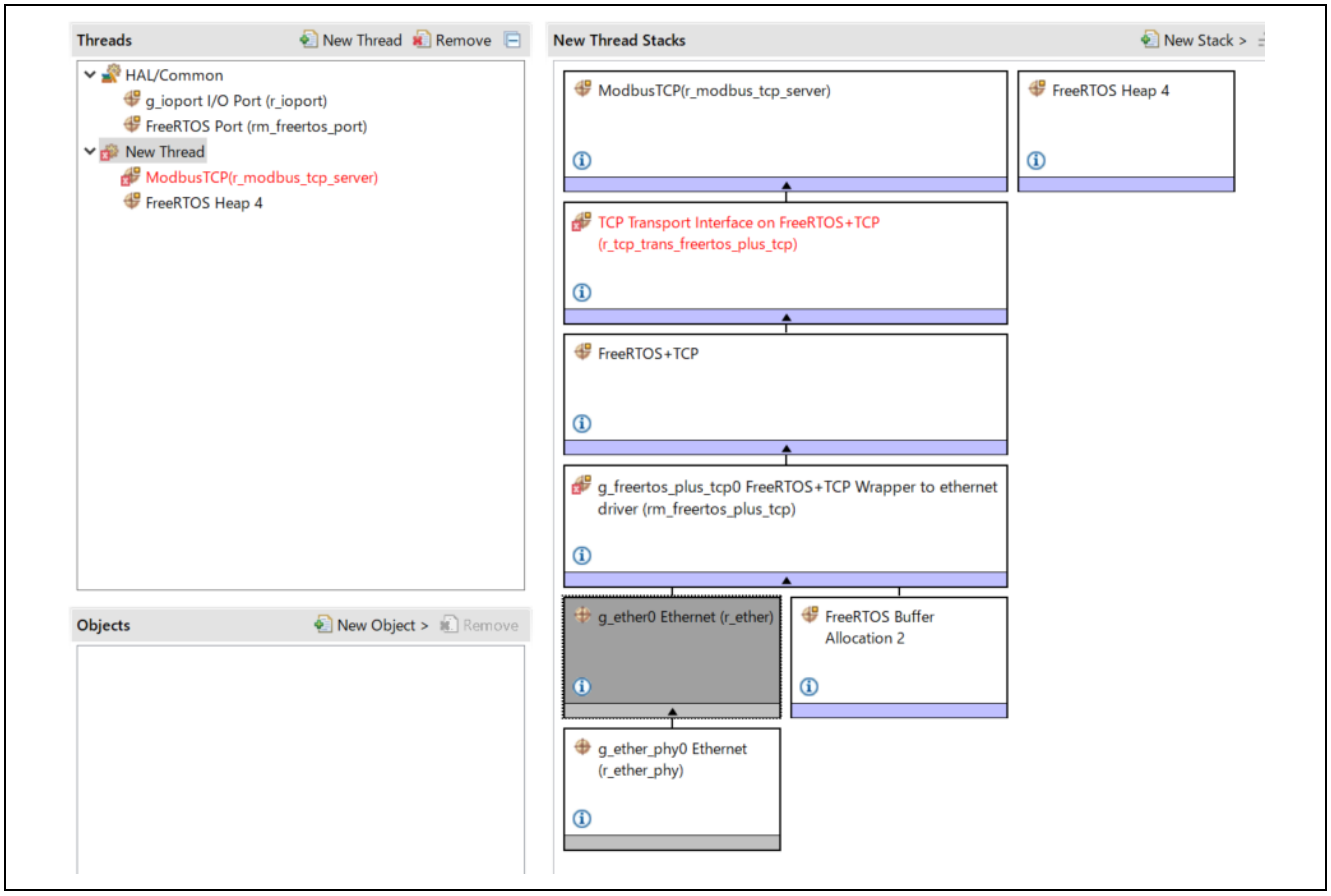
6.2.2 EK-RA6Mx / EK-RA8x1 / MCK-RA8T1 Creating Procedures

This section describes the procedures for creating EK-RA6Mx (EK-RA6M3, EK-RA6M4, EK-RA6M5) / EK-RA8x1 (EK-RA8D1, EK-RA8M1) / MCK-RA8T1.

1. Add Ethernet Driver
   Click “New” → “Ethernet (r_ether)” to “Add Ethernet Driver”.

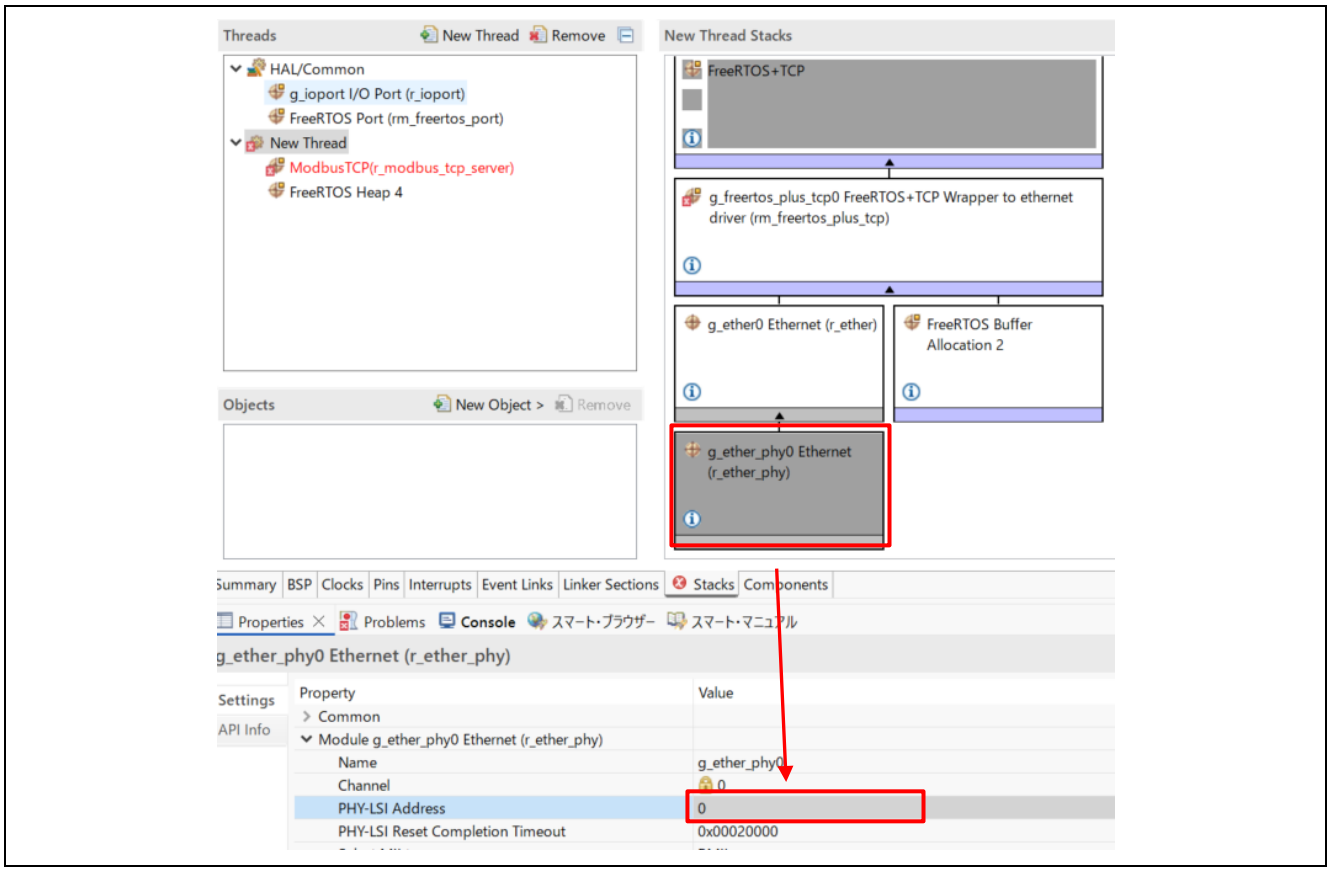


The stack will be configured as follows:



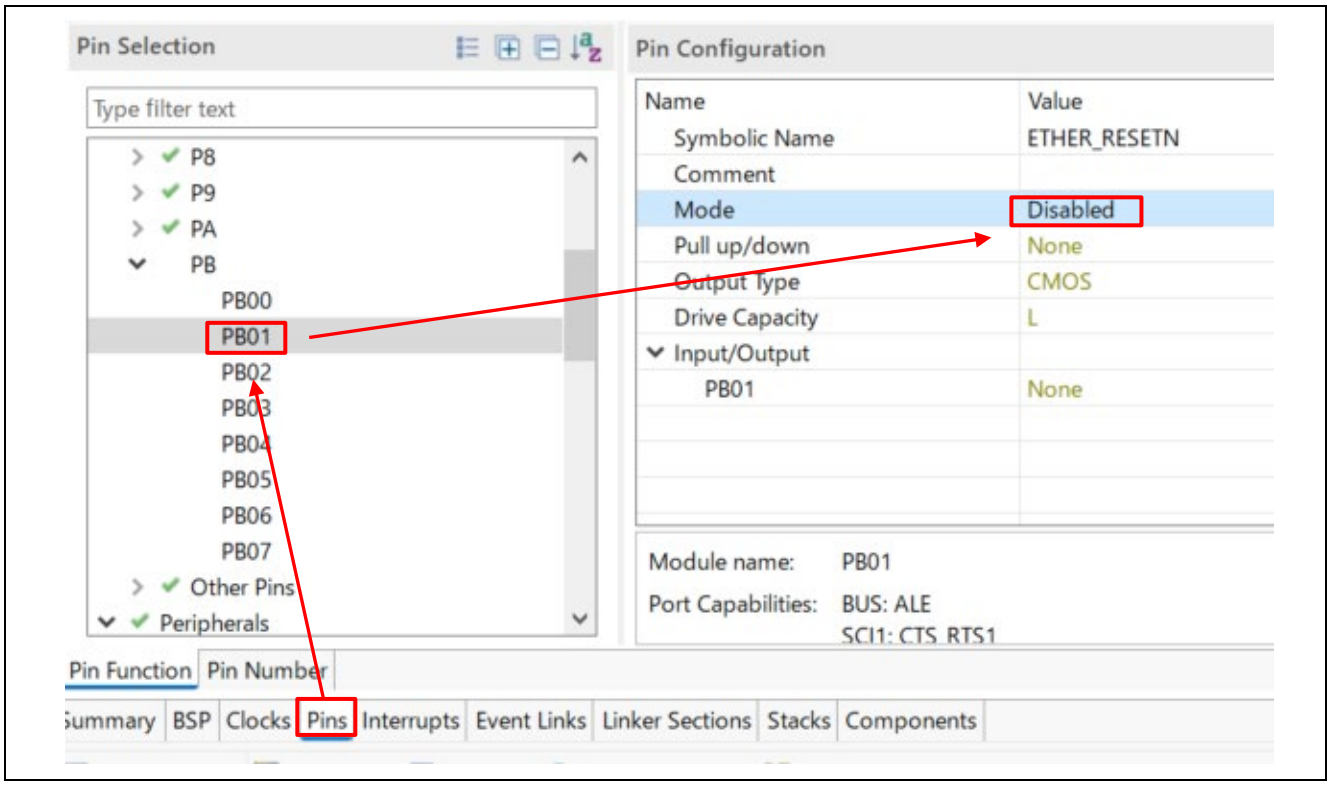
2. Set PHY Address
   Open “Properties” of “g_ether_phy0 Ethernet (r_ether_phy)” in “Stacks” and change “PHY-LSI Address” in “Module g_ether_phy0 Ethernet (r_ether_phy)” to the port0 address configured on the evaluation board.

   RA6xx：PHY Address **：0**, RA8xx：PHY Address：**5**

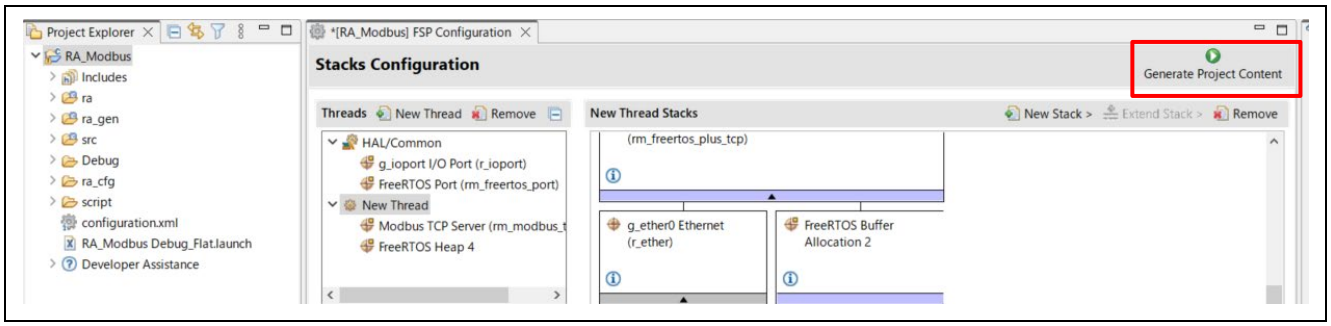


3. Set PHY Reset pin
The setting of the PHY Reset terminal on the evaluation boards RA8D1 and RA8T1 needs to be changed.
Open “Pin Configuration” of [Reset pin for evaluation board] (see below) in “Pins” and change “Mode” to “Disabled”.

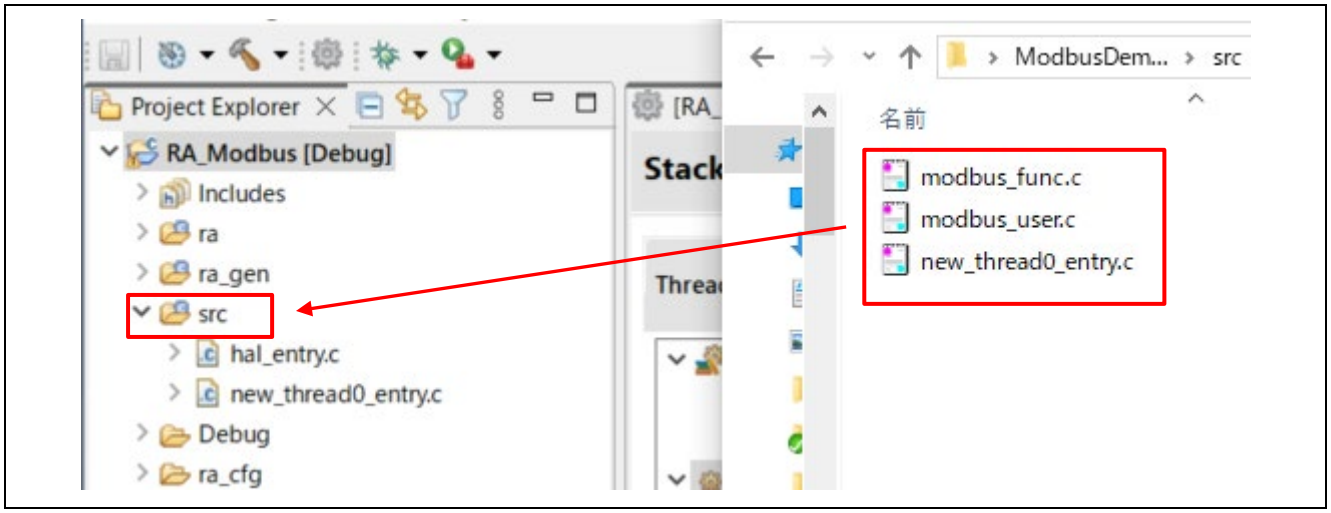
RA8D1 : **P706**, RA8T1 : **PB01**
Note: This procedure is not required for evaluation boards other than those listed above.

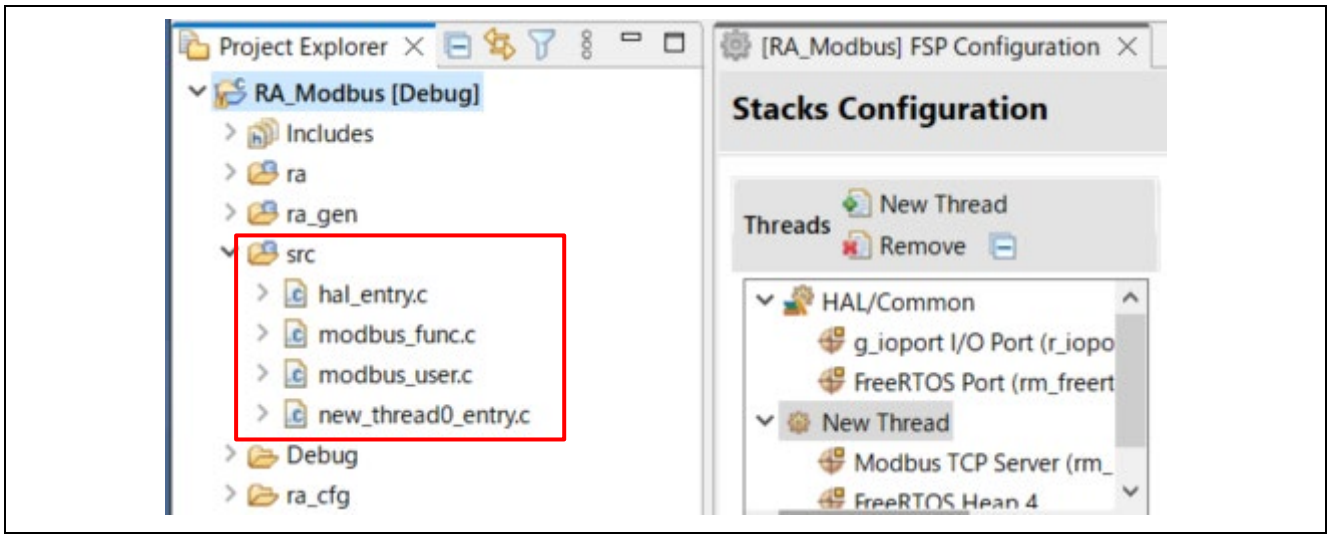


4. Generate the code
Generate the code with "Generate Project Content".



5. Add Modbus Sample Application
   Copy modbus_func.c, modbus_user.c, and new_thread0_entry.c from the src folder of the sample program package to the src folder of the project and overwrite them.

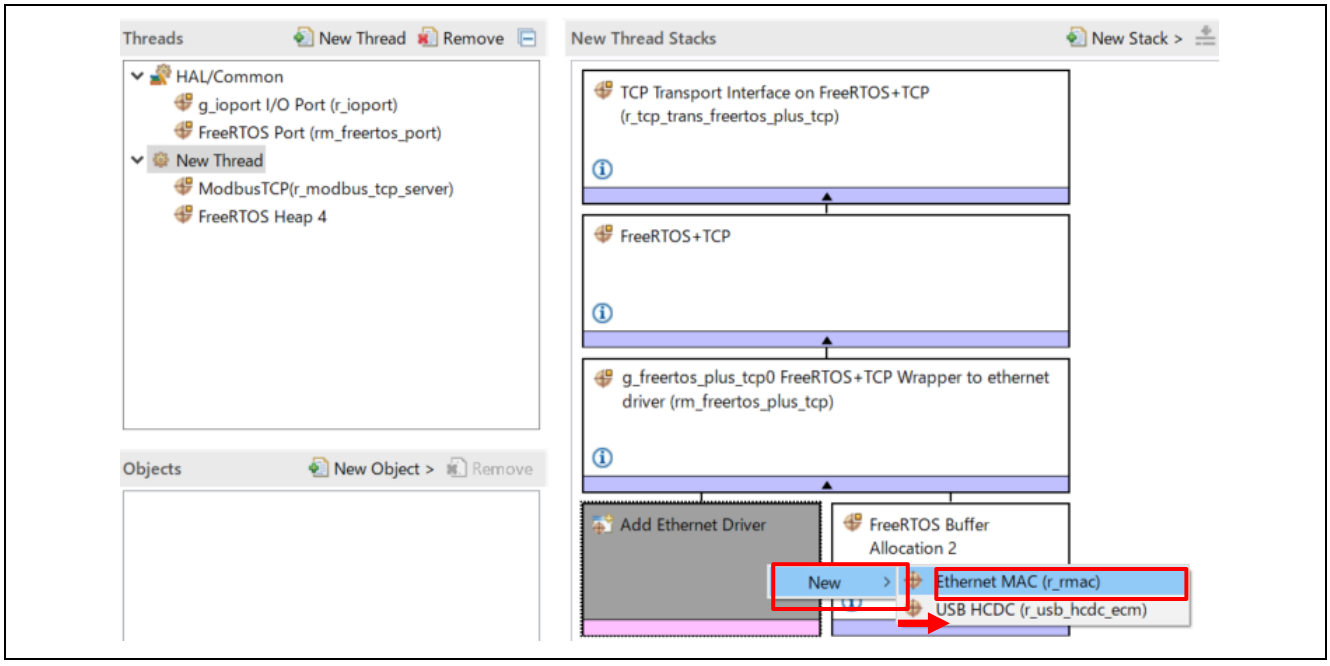




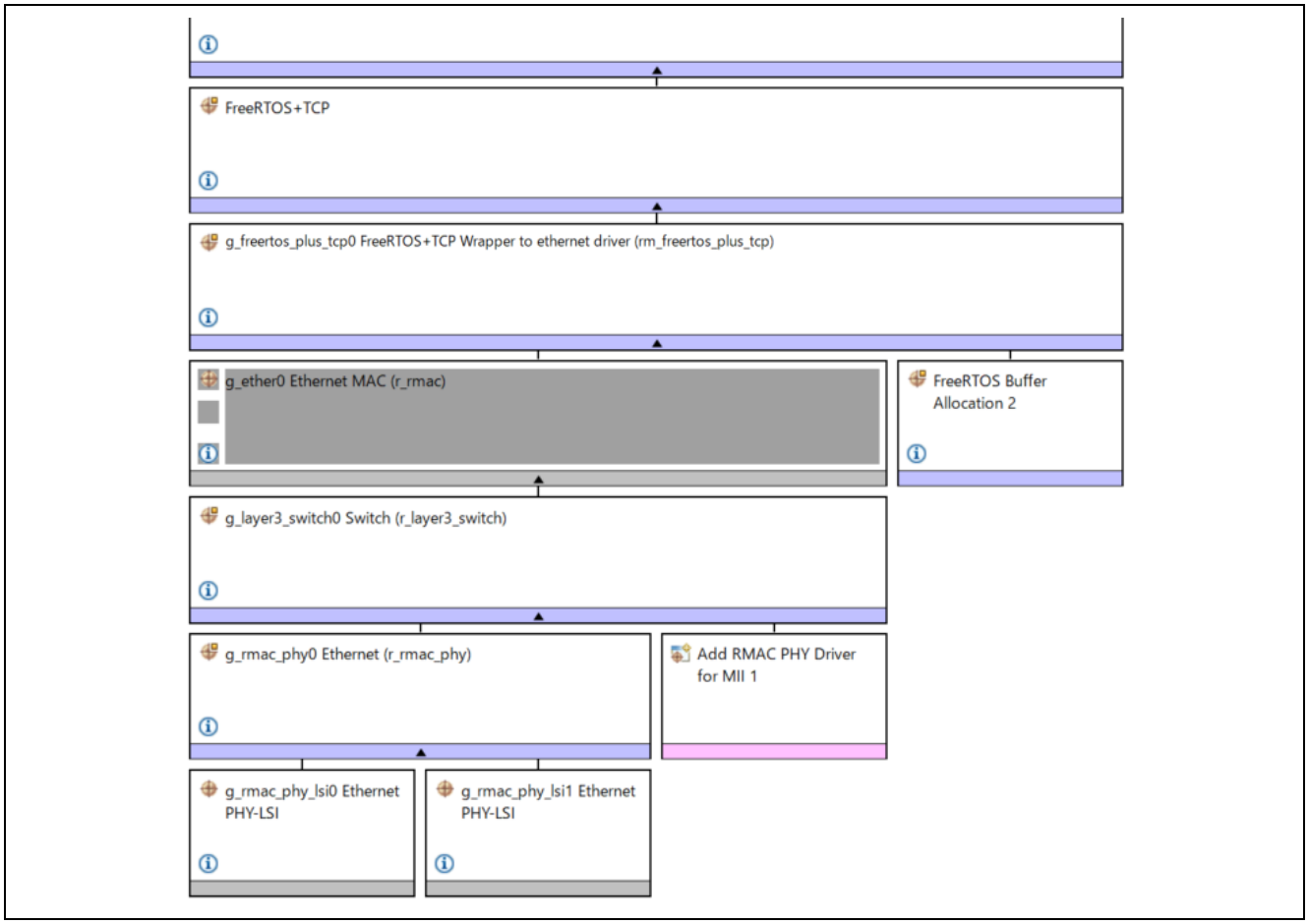
6.2.3 MCK-RA8T2 Creating Procedures

This section describes the procedures for creating MCK-RA8T2.

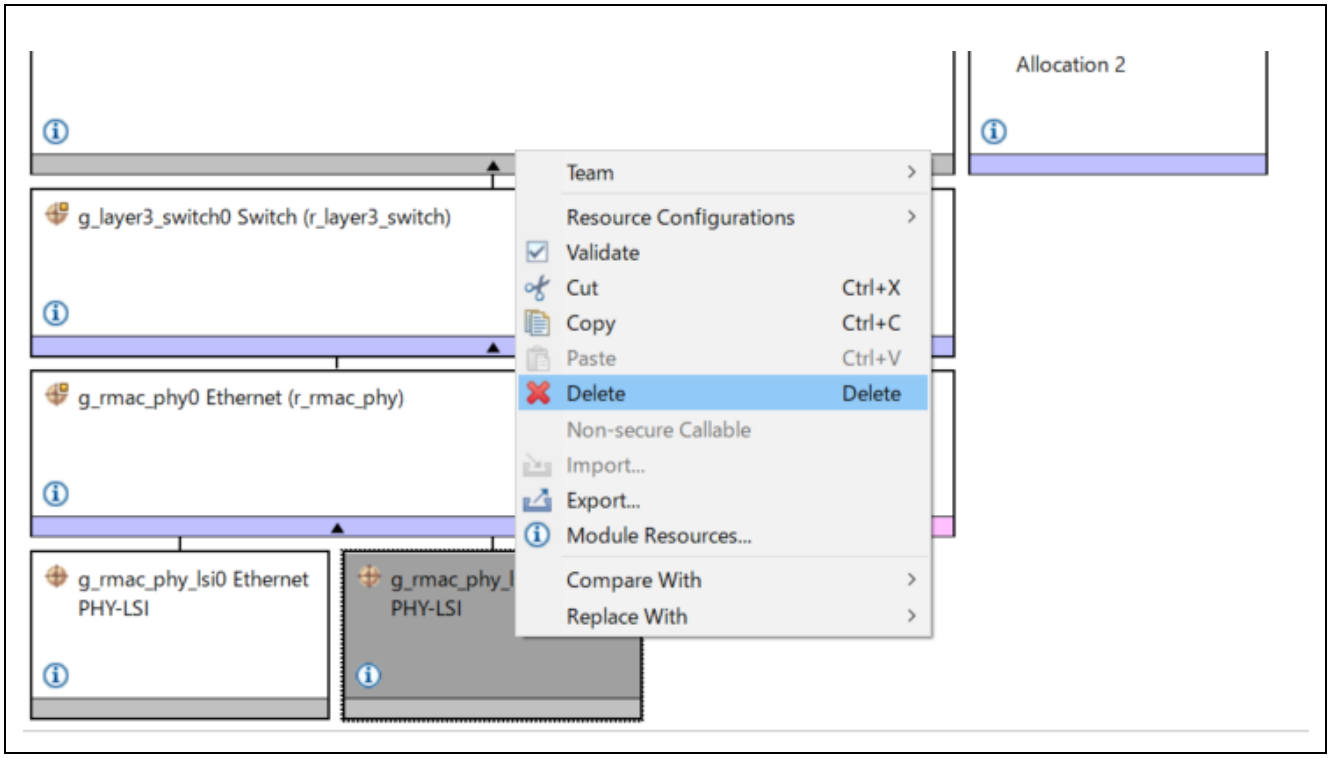
1. Add Ethernet Driver
   Click “New” → “Ethernet (r_rmac)” to “Add Ethernet Driver”.



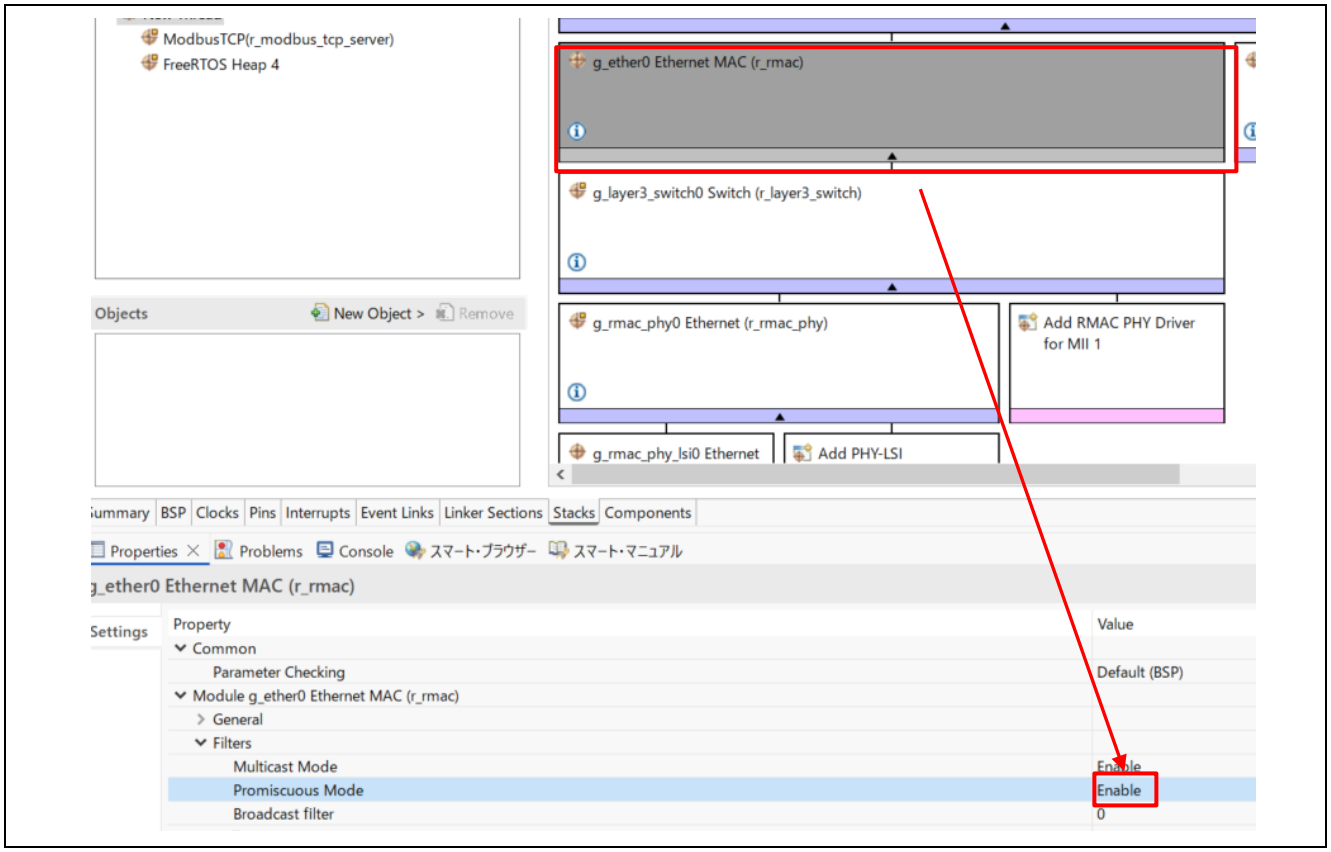
The stack will be configured as follows:



Right-click on “g_rmac_phy_lsi1 Ethernet PHY-LSI”, select Delete, and then click “OK” on the “Remove Stack Elements” pop-up.

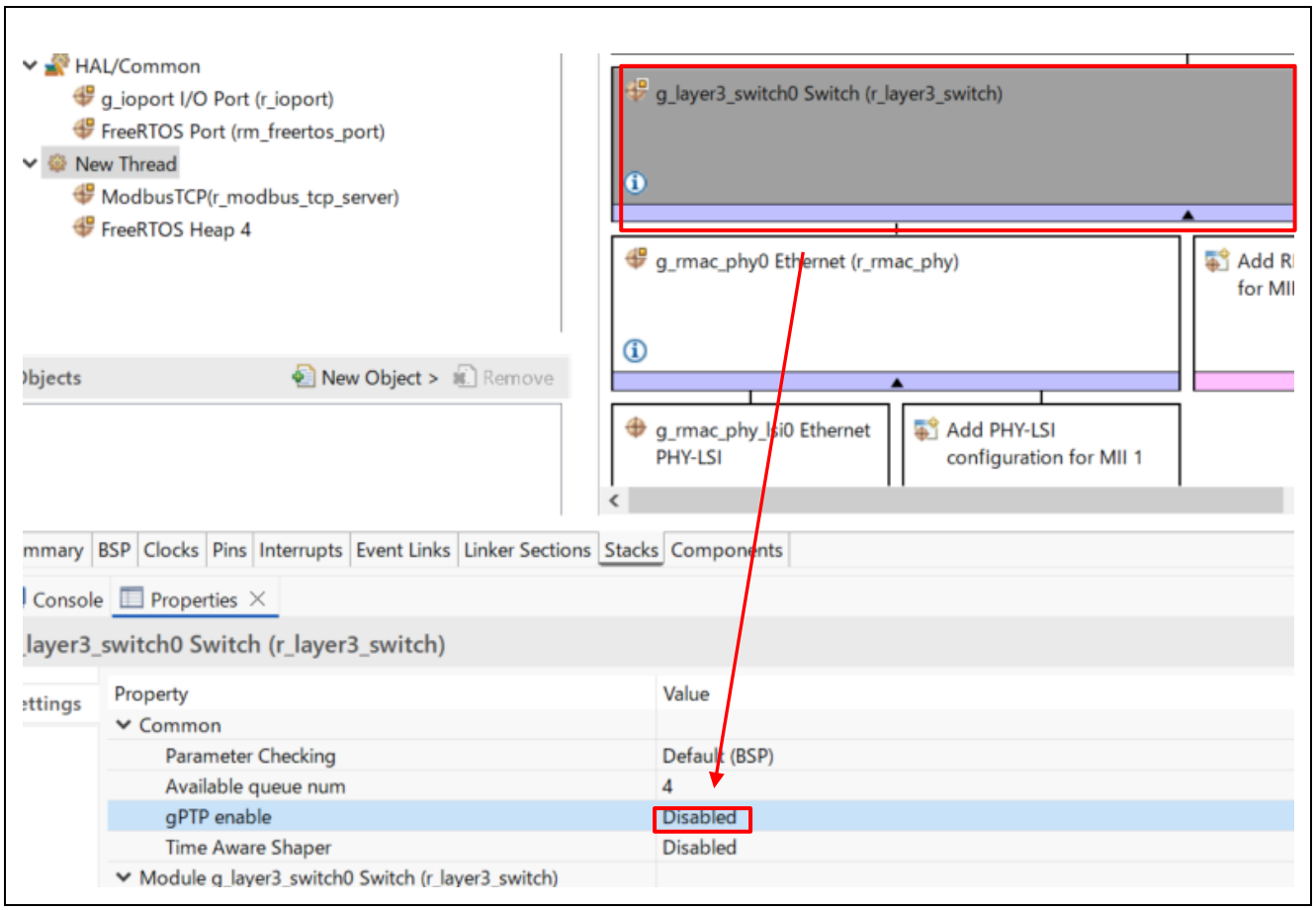


2. Set r_rmac
Change the “Stacks” → ”g_ether0 Ethernet MAC (r_rmac)” → ”Module g_ether0 Ethernet MAC (r_rmac)” → “Promiscuous Mode” in “Filters” to “Enable”.



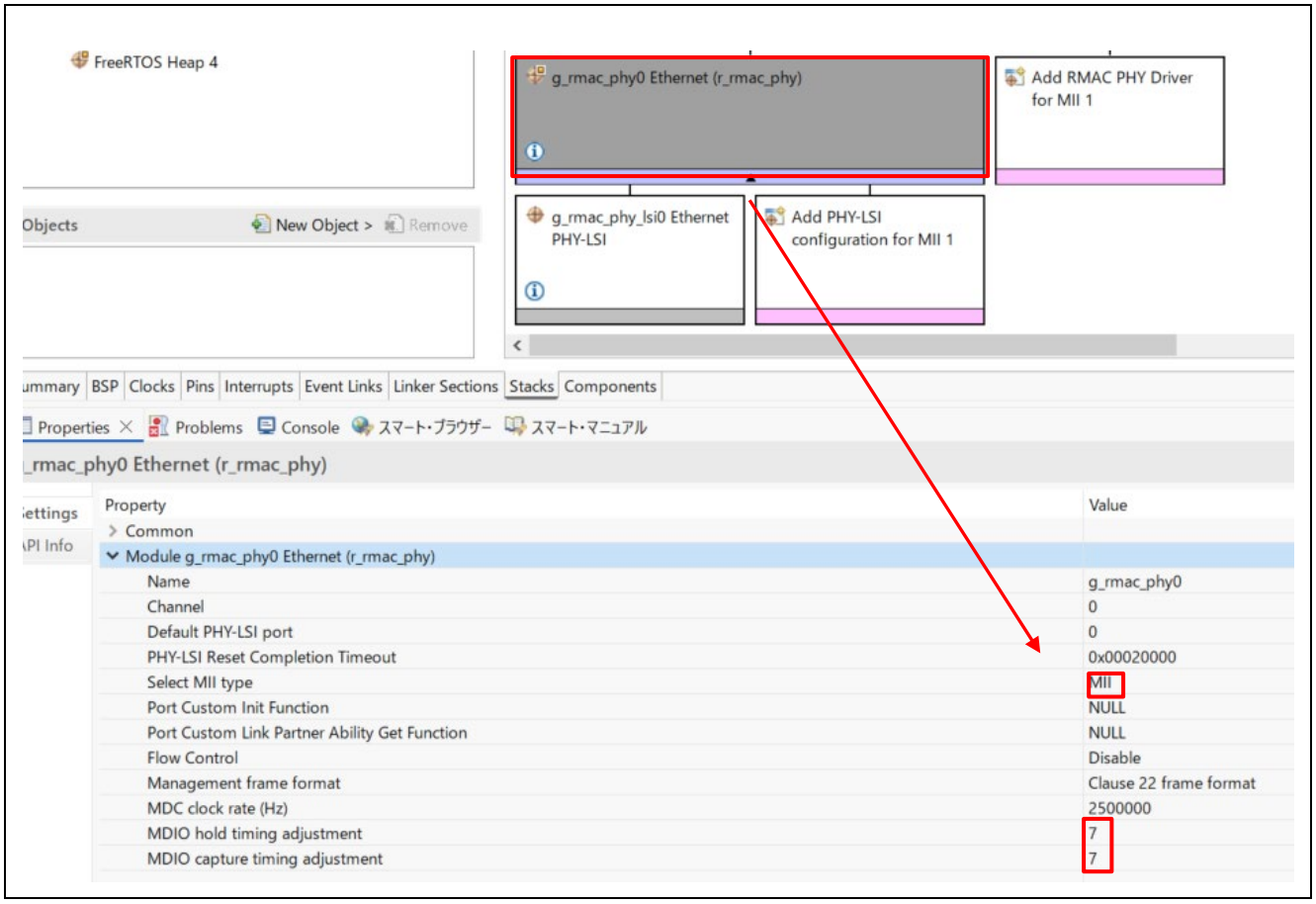
3. Set r_layer3_switch
   Open “Properties” of “g_layer3_switch0 Switch (r_layer3_switch)” in “Stacks” and change “gPTP enable” to the following values.

   gPTP enable：**Disabled**

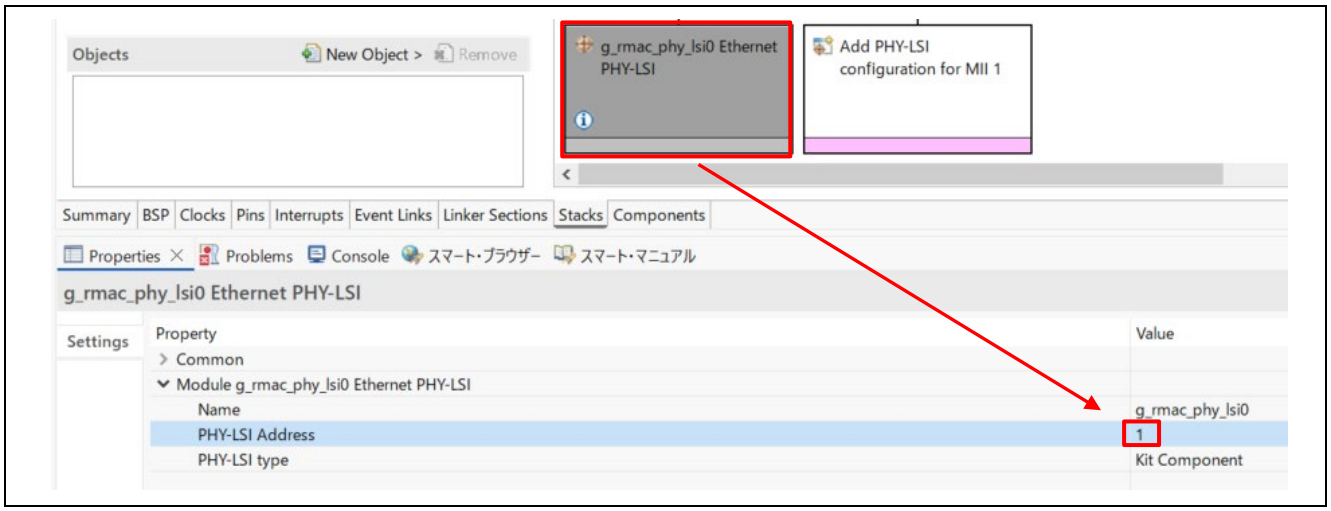


4. Set r_rmac_phy
   Open “Properties” of “g_mac_phy0 Ethernet (r_rmac_phy)” in “Stacks” and change “Select MII type”, “MDIO hold timing adjustment”, and “MDIO capture timing adjustment” in “Module g_mac_phy0 Ethernet (r_rmac_phy)” to the following values.

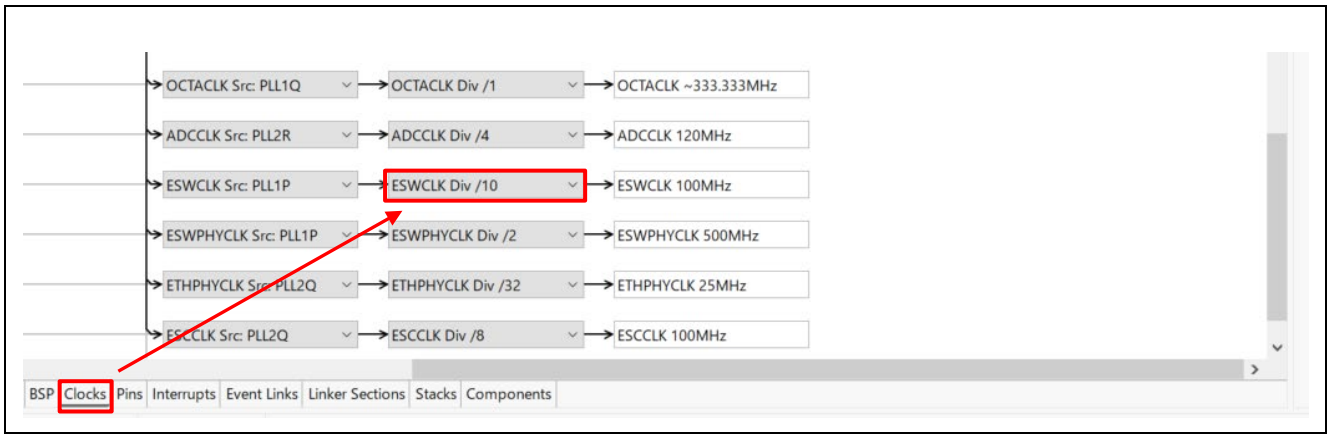
   Select MII type：**MII**
   MDIO holds timing adjustment：**7**
   MDIO capture timing adjustment：**7**



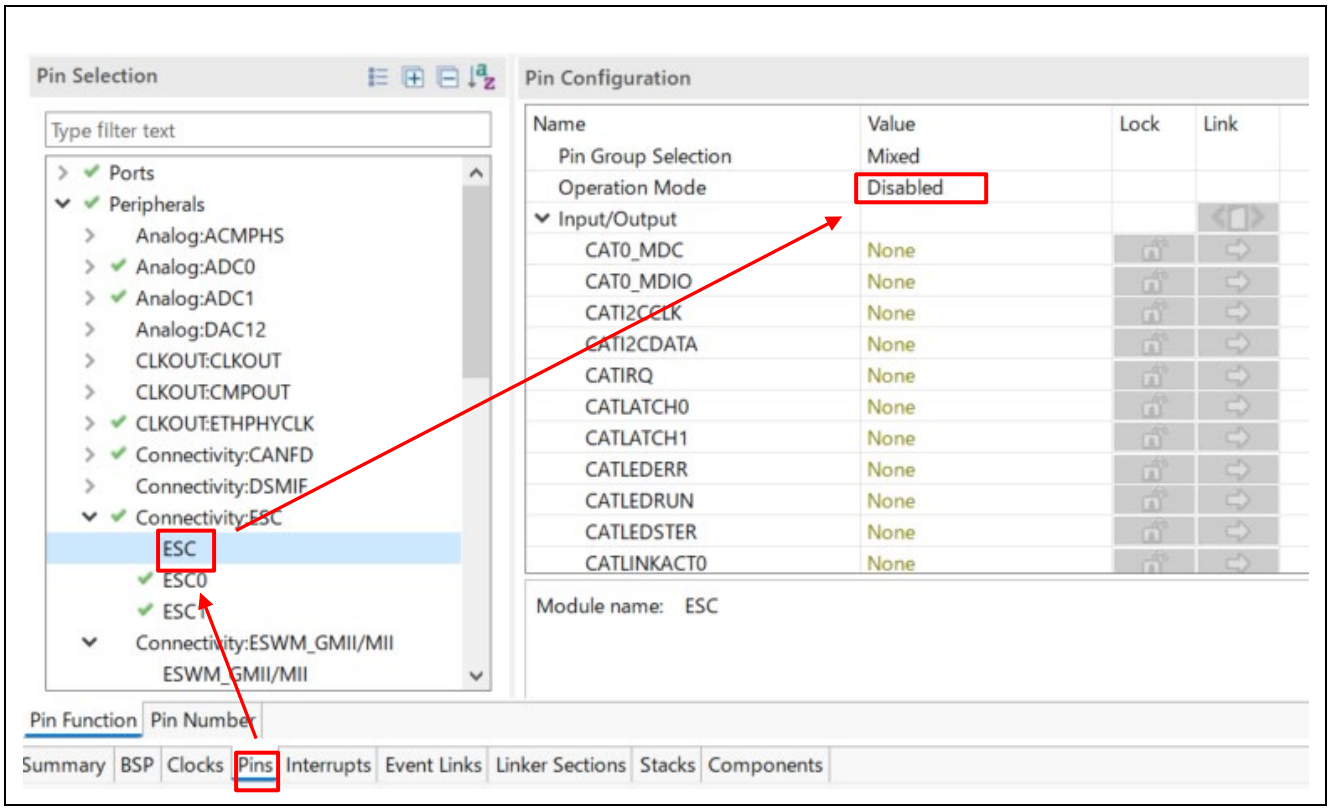
5. Set PHY Address
   Open “Properties” of “g_mac_phy_lsi0 Ethernet PHY LSI” in “Stacks” and change “PHY-LSI Address” in “Module g_mac_phy_lsi0 Ethernet PHY LSI” to “1”.



6. Set ESW Clock
Change the “ESWCLK Div” at the bottom of “Clocks” to “ESWCLK Div /10”.

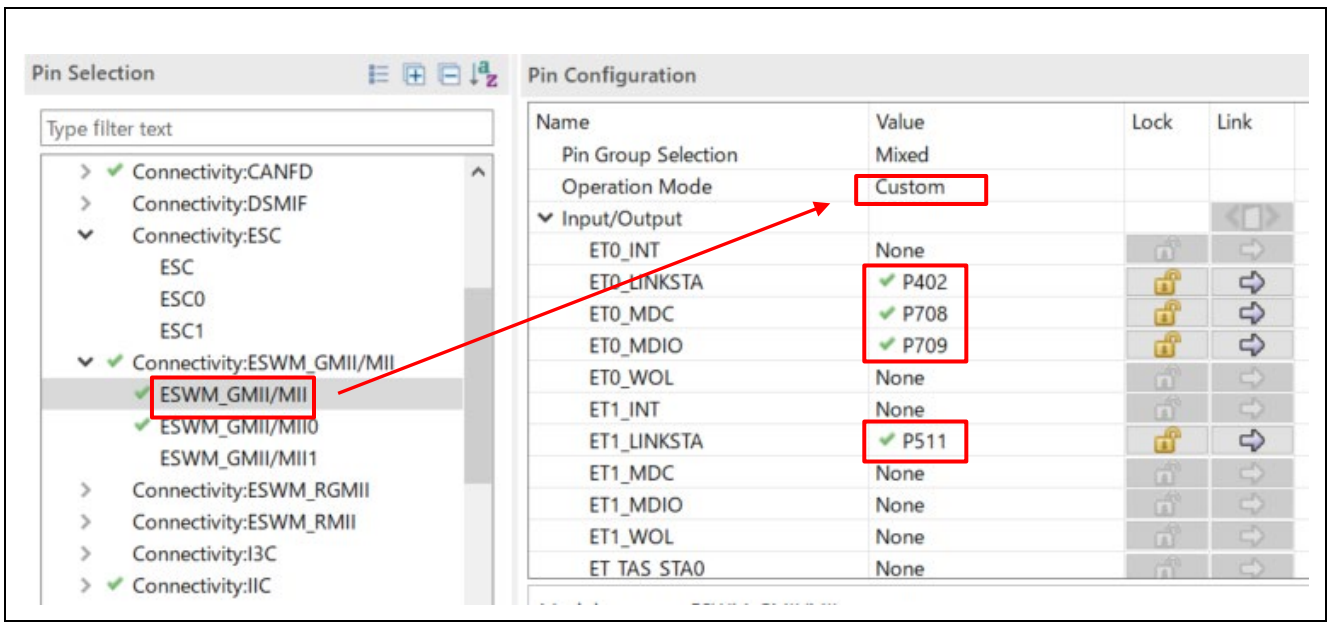


7. Set ESWM Terminals
   Change the "Operation Mode" of “Pins” → ”Peripherals” → ”Connectivity: ESC” → "ESC" to "Disabled".
   Change the "Operation Mode" of "ESC0" and "ESC1" to "Disabled" as well.



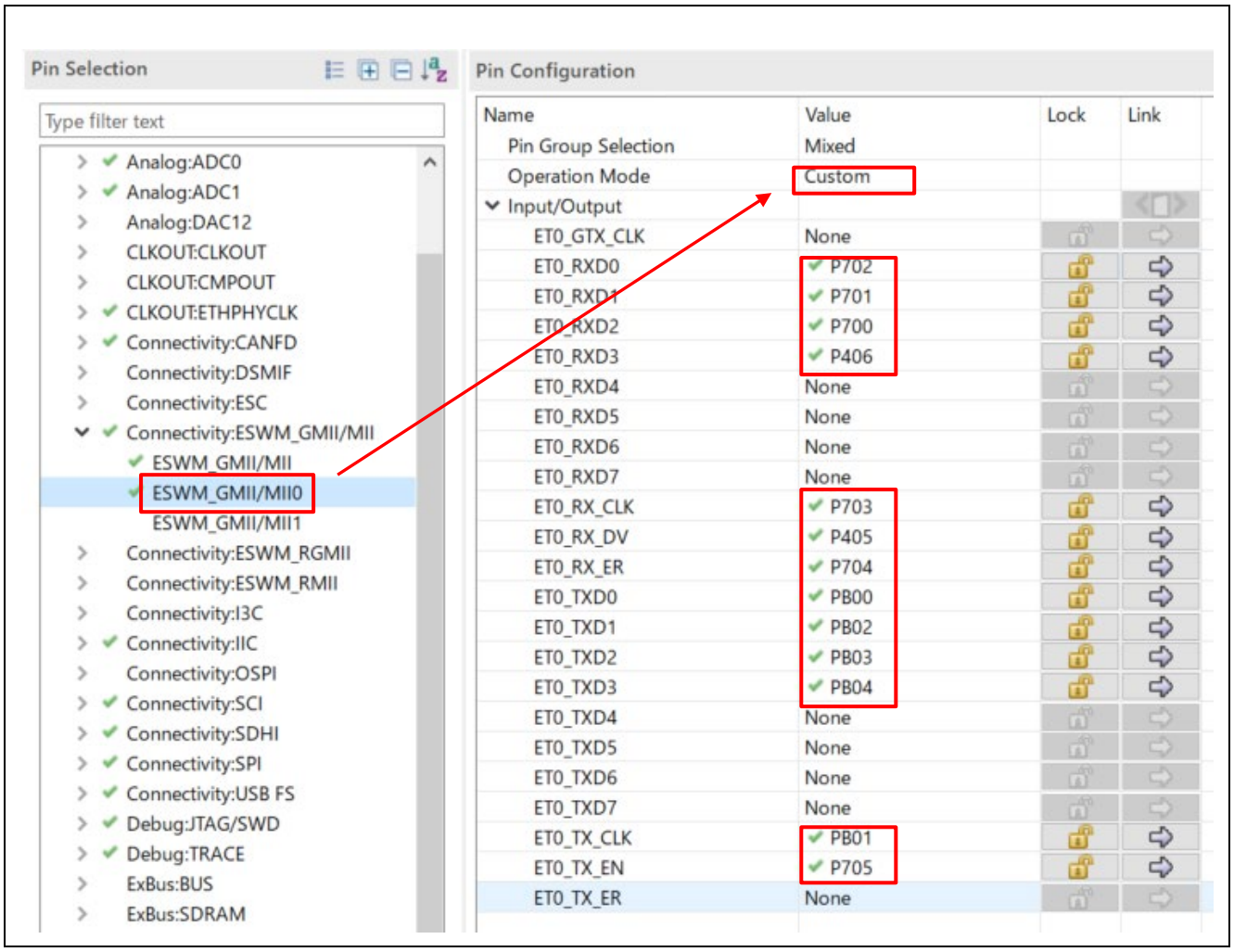
After changing the "Operation Mode" of "Pins" → "Peripherals" → "Connectivity: ESWM_GMII/MII" → "ESWM_GMII/MII" to "Custom", change the pins as follows:

ET0_LINKSTA : **P402**
ET0_MDC : **P708**
ET0_MDIO : **P709**
ET1_LINKSTA : **P511**

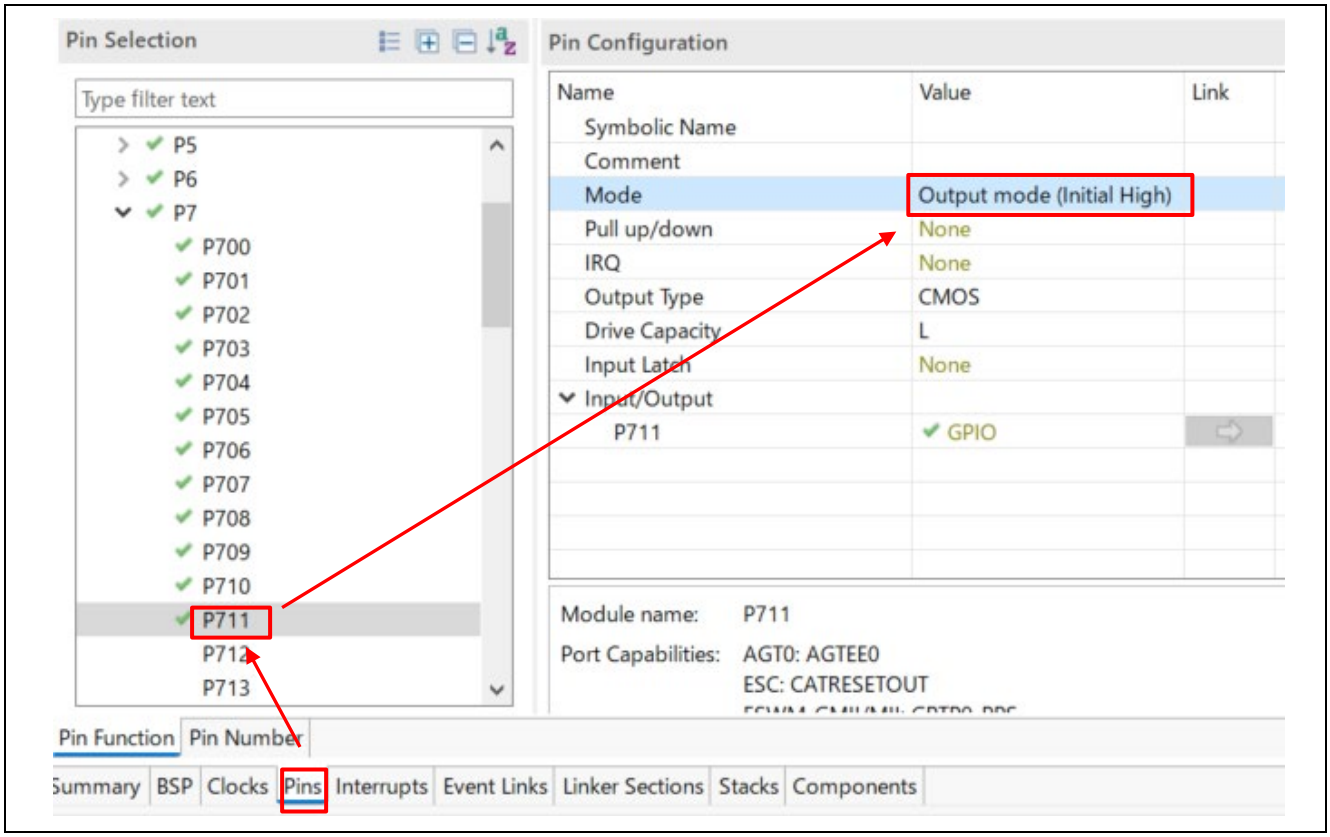


After changing the "Operation Mode" of "Pins" → "Peripherals" → "Connectivity: ESWM_GMII/MII" → "ESWM_GMII/MII0" to "Custom", change the pins as follows:

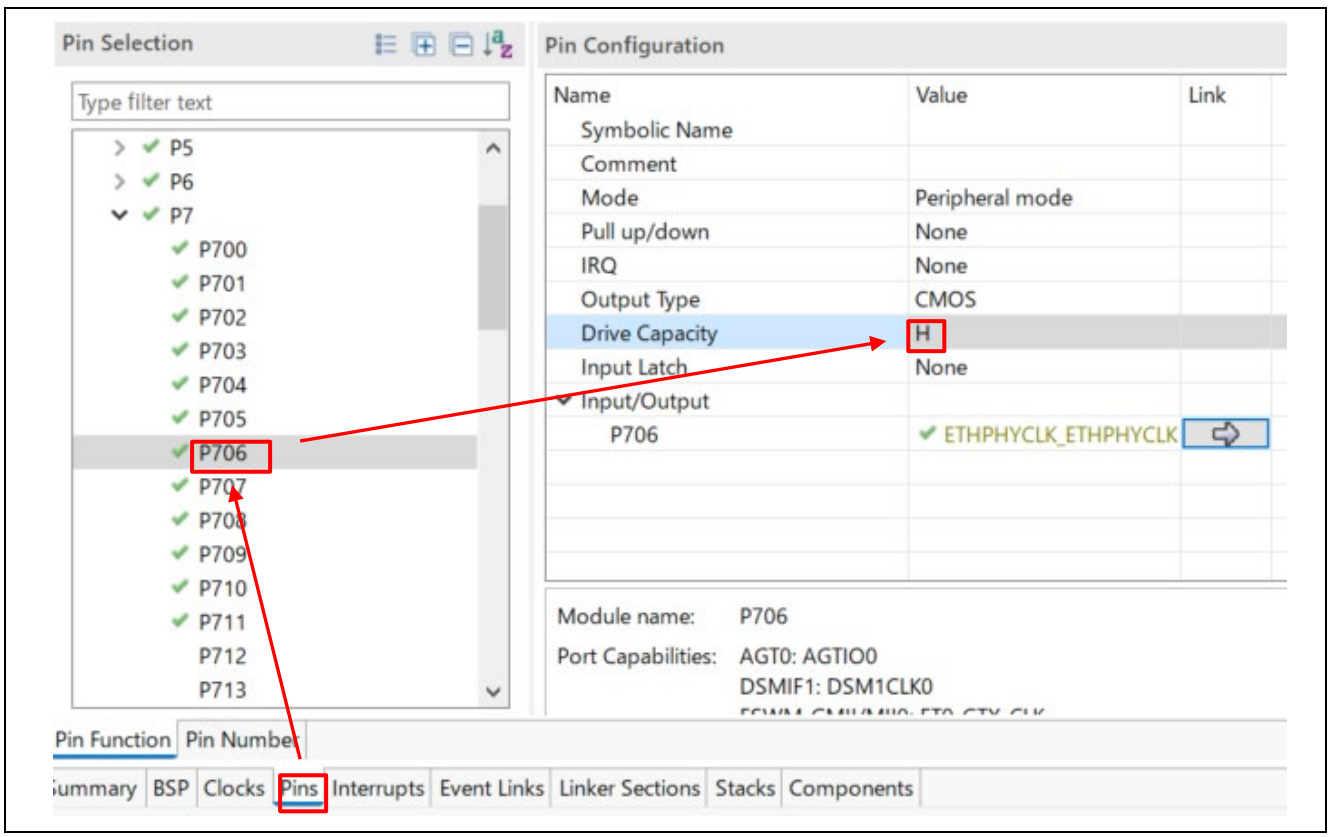
ET0_RXD0 : **P702**
ET0_RXD1 : **P701**
ET0_RXD2 : **P700**
ET0_RXD3 : **P406**
ET0_RX_CLK : **P703**
ET0_RX_DV : **P405**
ET0_RX_ER : **P704**
ET0_TXD0 : **PB00**
ET0_TXD1 : **PB02**
ET0_TXD2 : **PB03**
ET0_TXD3 : **PB04**
ET0_TX_CLK : **PB01**
ET0_TX_EN : **P705**



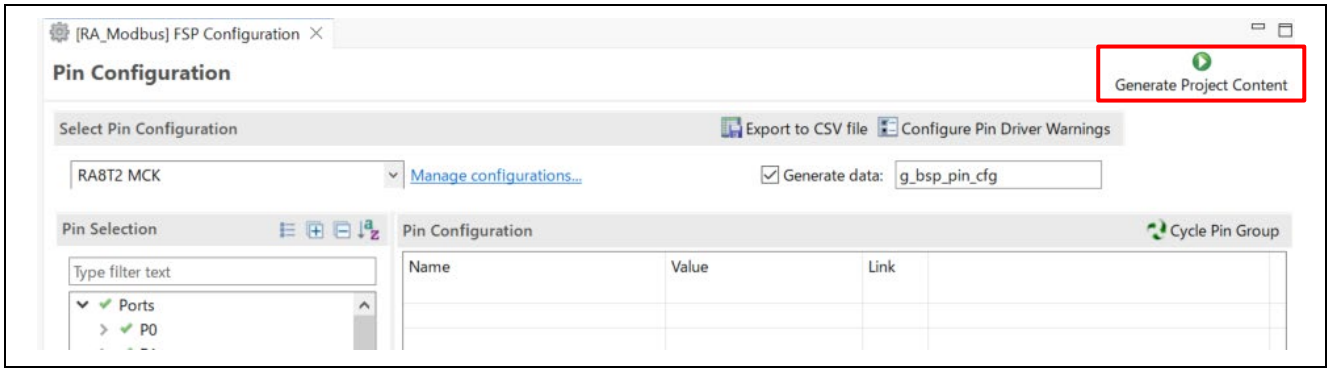
8. Set PHY Reset Terminals
   Change the "Mode" of “Pins” → ”Ports” → ”P7” → "P711" to "Output mode (Initial High)".



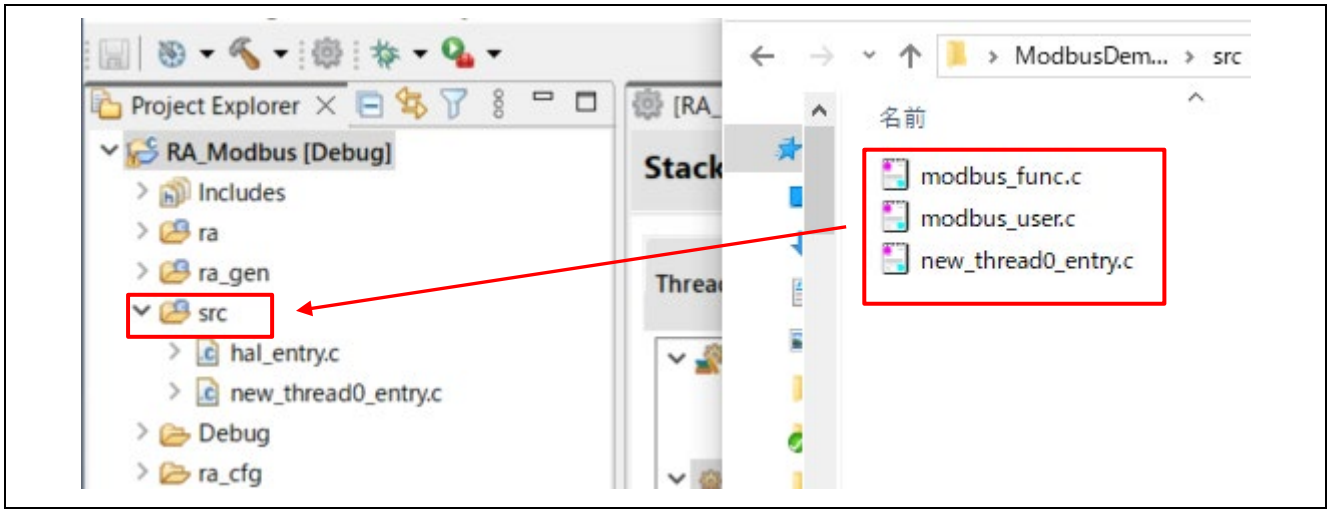
9. Set PHY CLK
   Change the "Drive Capacity" of “Pins” → ”Ports” → ”P7” → "P706" to "H".

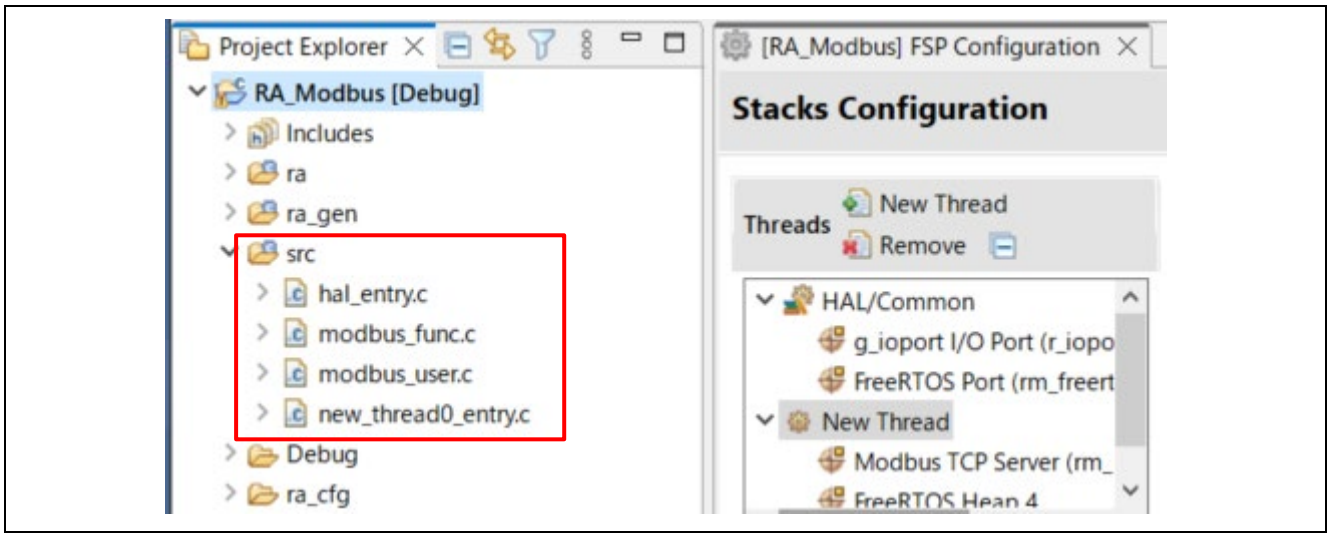


10. Generate the code
    Generate the code with "Generate Project Content".



11. Add Modbus Sample Application
    Copy modbus_func.c, modbus_user.c, and new_thread0_entry.c from the src folder of the sample program package to the src folder of the project and overwrite them.

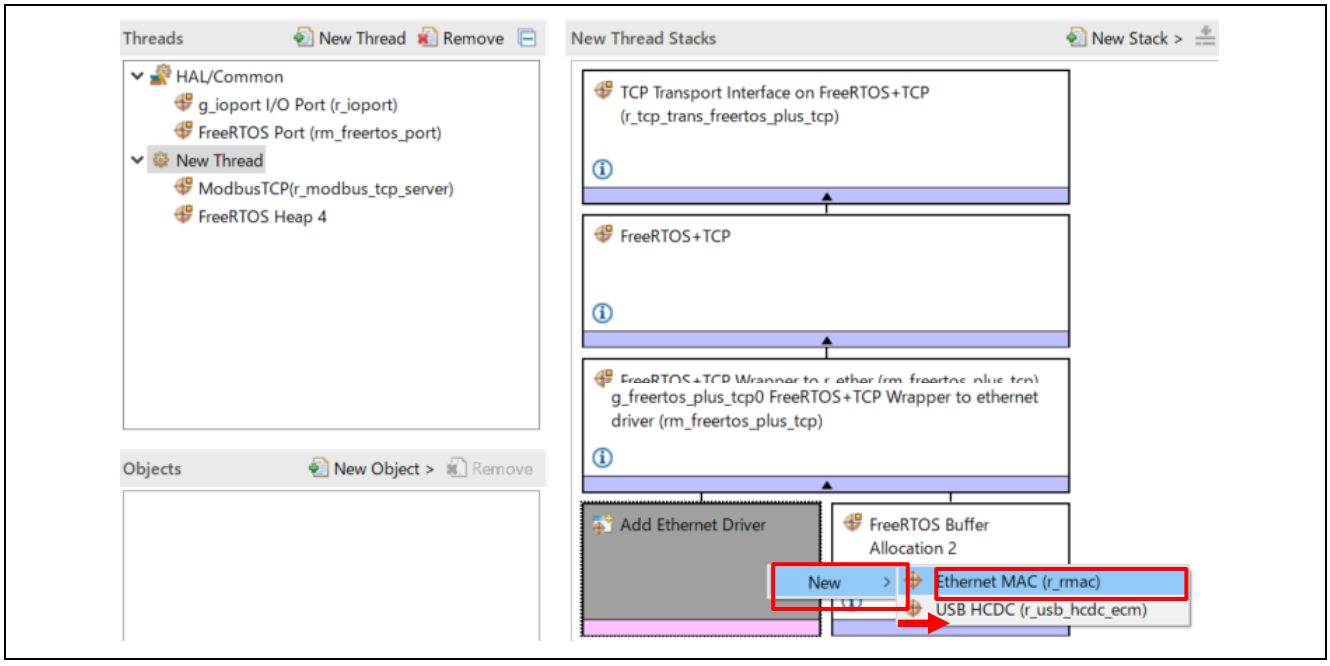




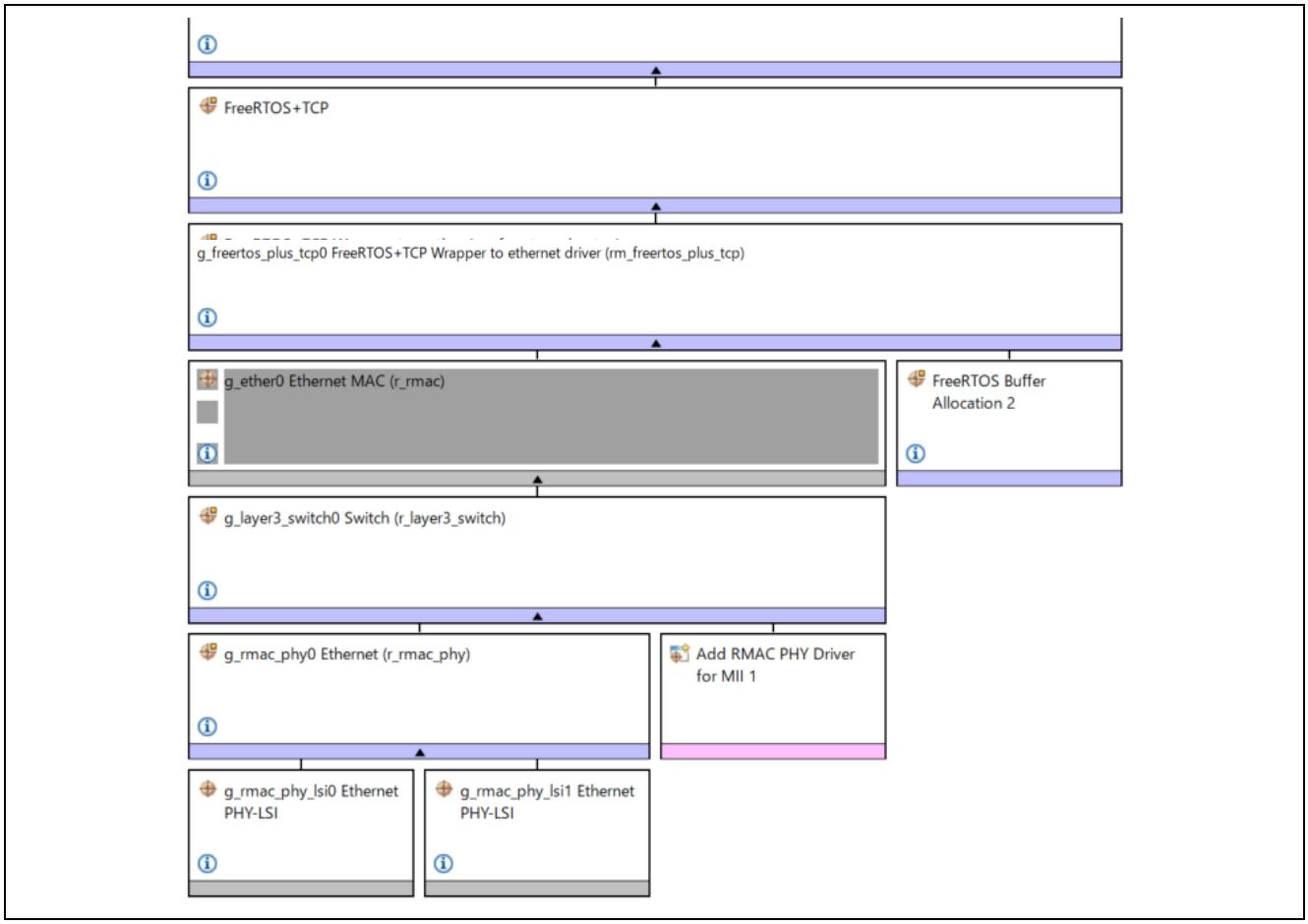
6.2.4 EK-RA8D2 Creating Procedures

This section describes the procedures for creating EK-RA8D2.

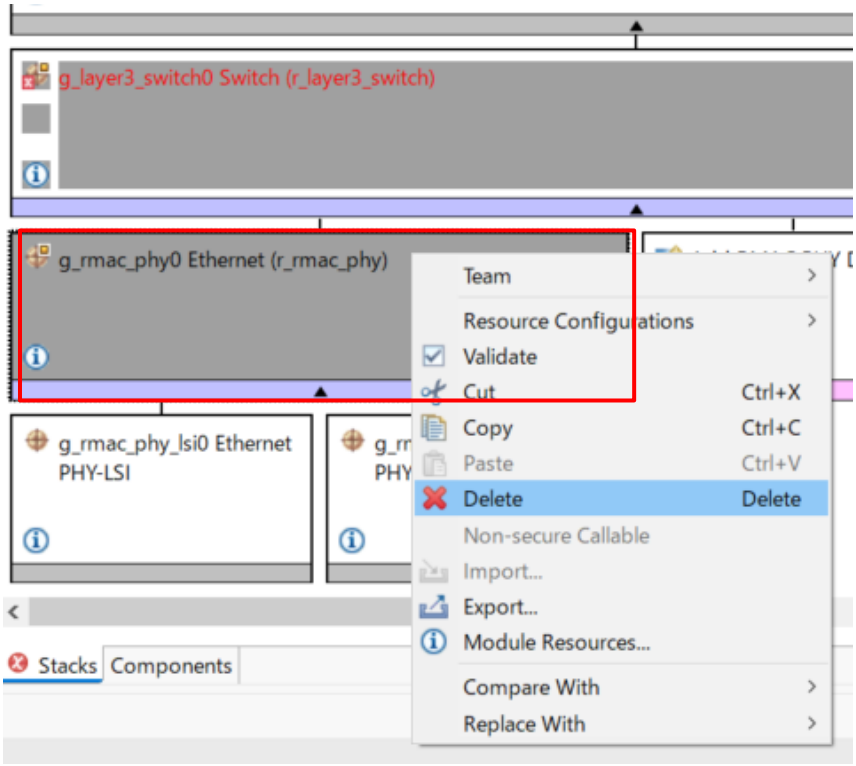
1. Add Ethernet Driver
   Click “New” → “Ethernet (r_rmac)” to “Add Ethernet Driver”.



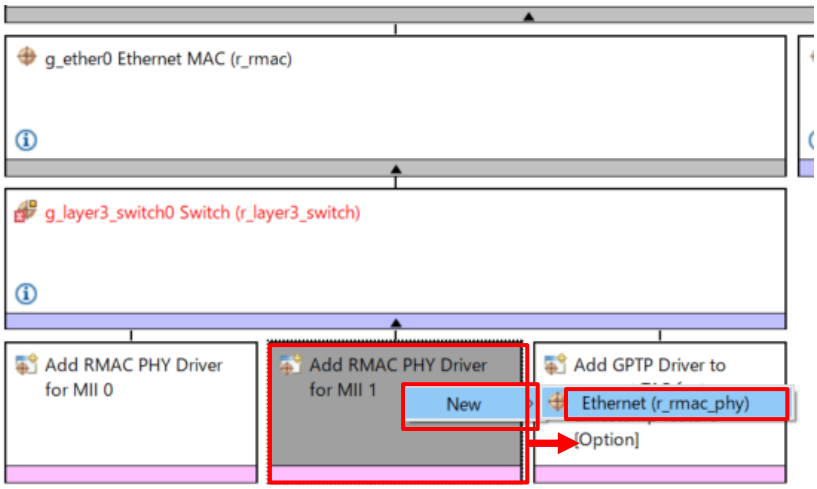
The stack will be configured as follows:



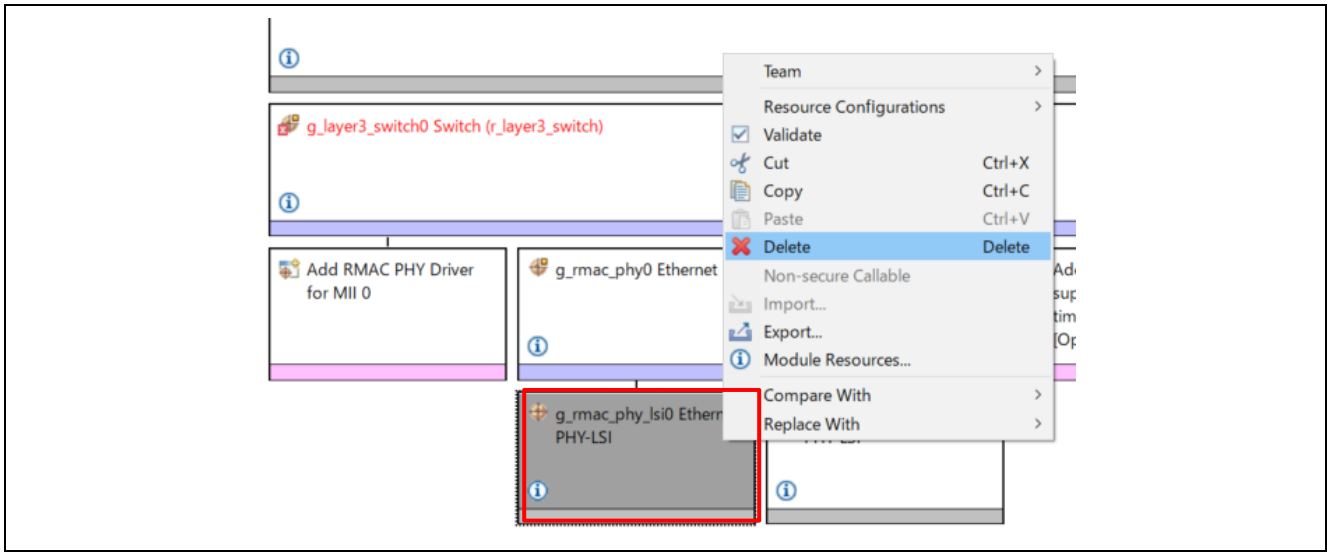
Right-click on “g_mac_phy0 Ethernet (r_rmac_phy)”, select Delete, and then click “OK” on the “Remove Stack Elements” pop-up.



Click “New” → “Ethernet (r_rmac_phy)” to “Add RMAC PHY Driver for MII1”.

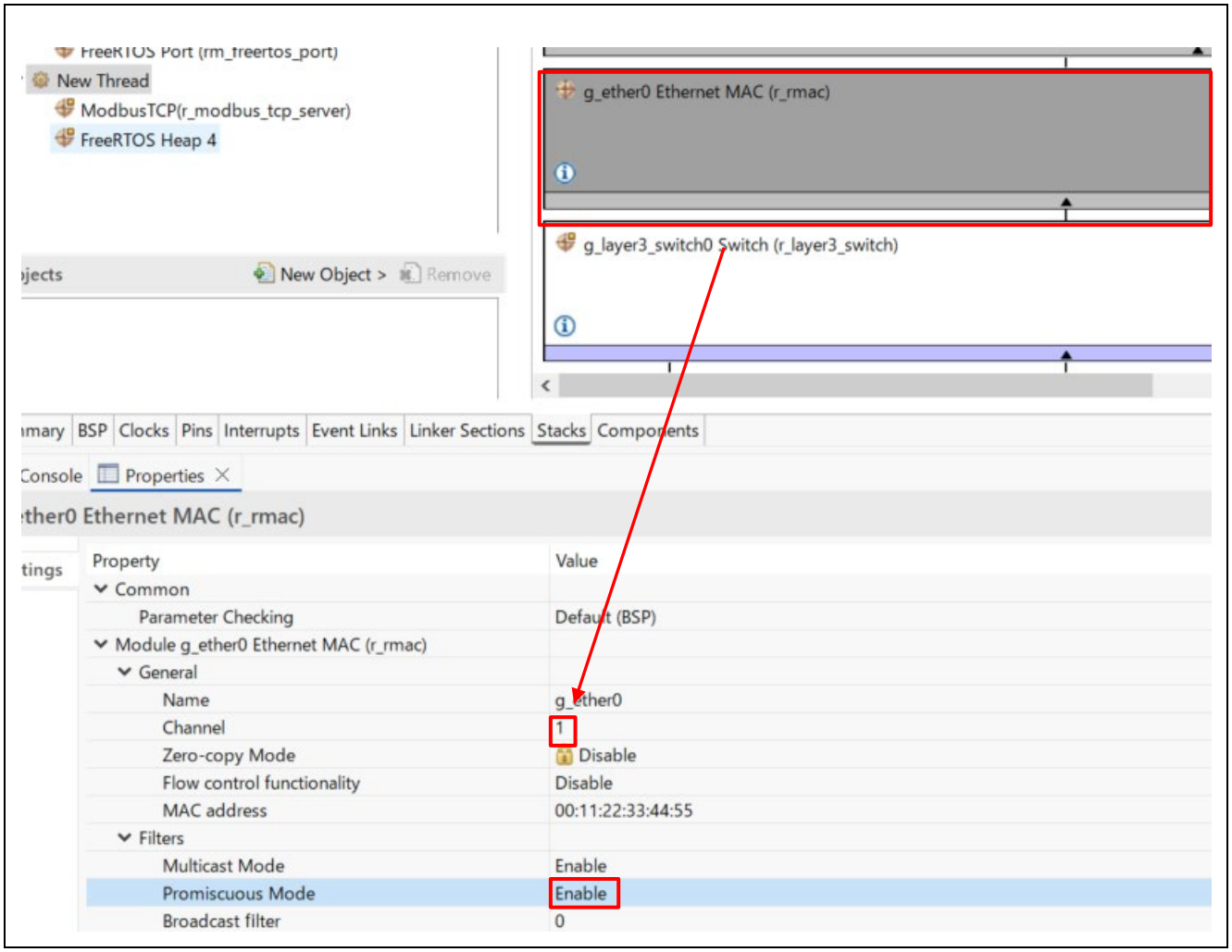


Right-click on “g_rmac_phy_lsi0 Ethernet PHY-LSI”, select Delete, and then click “OK” on the “Remove Stack Elements” pop-up.



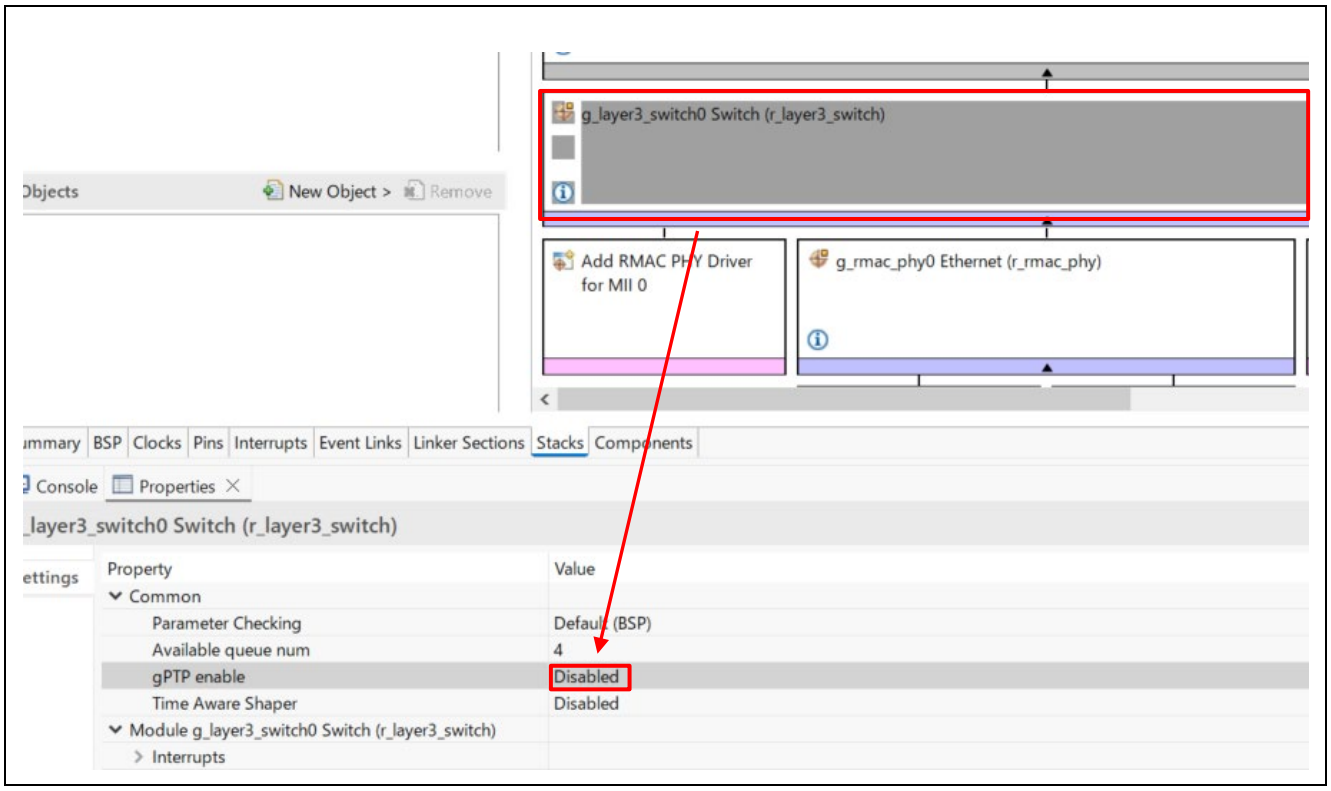
2. Set r_rmac
   Open “Properties” of “g_ether0 Ethernet MAC (r_rmac)” in “Stacks” and change “Channel” and “Promiscuous Mode” in “Module g_mac_phy0 Ethernet (r_rmac_phy)” to the following values.

   Channel：**1**
   Promiscuous Mode：**Enable**



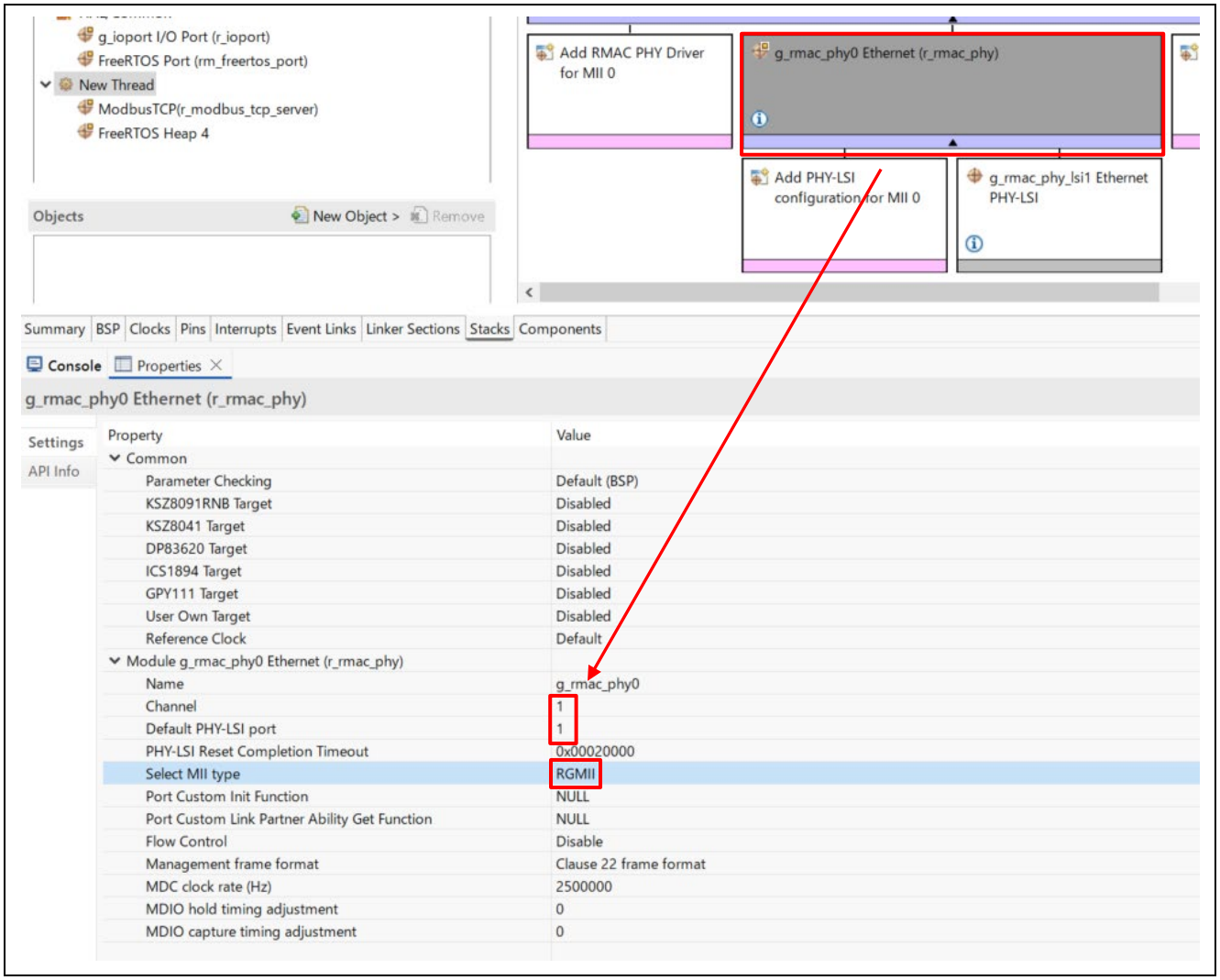
3. Set r_layer3_switch
   Open “Properties” of “g_layer3_switch0 Switch (r_layer3_switch)” in “Stacks” and change “gPTP enable” to the following values.

   gPTP enable：**Disabled**

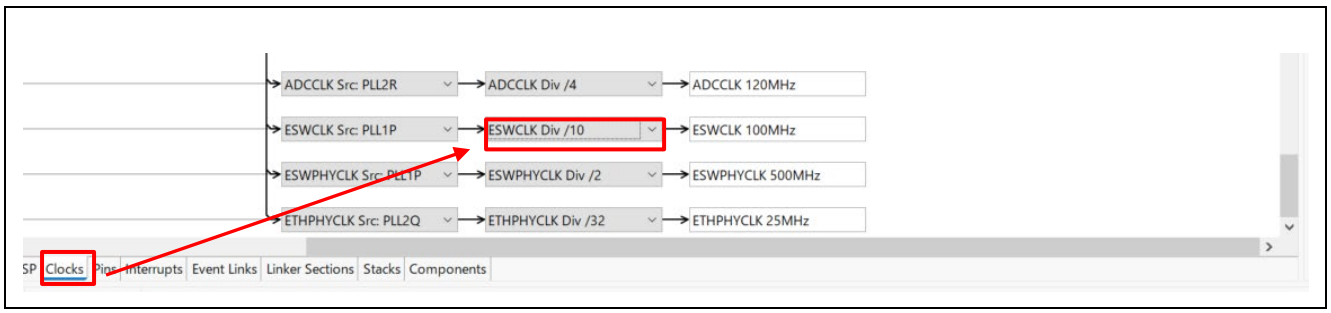


4. Set r_rmac_phy
   Open “Properties” of “g_mac_phy0 Ethernet (r_rmac_phy)” in “Stacks” and change “Channel”, “Default PHY-LSI port” and “Select MII type” in “Module g_mac_phy0 Ethernet (r_rmac_phy)” to the following values.

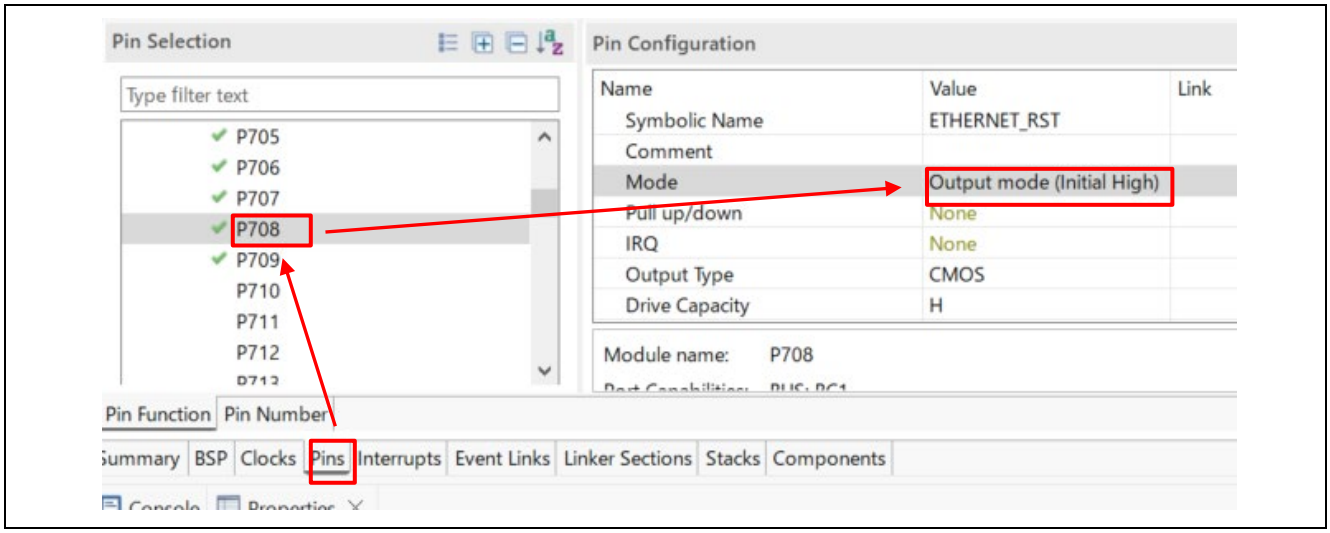
   Channel：**1**
   Default PHY-LSI port：**1**
   Select MII type：**RGMII**



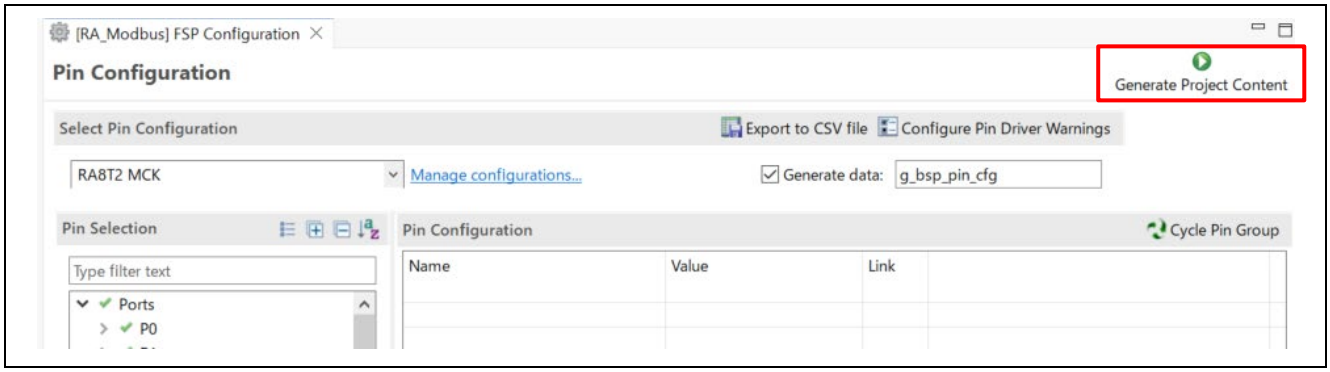
5. Set ESW Clock
   Change the “ESWCLK Div” at the bottom of “Clocks” to “ESWCLK Div /10”.



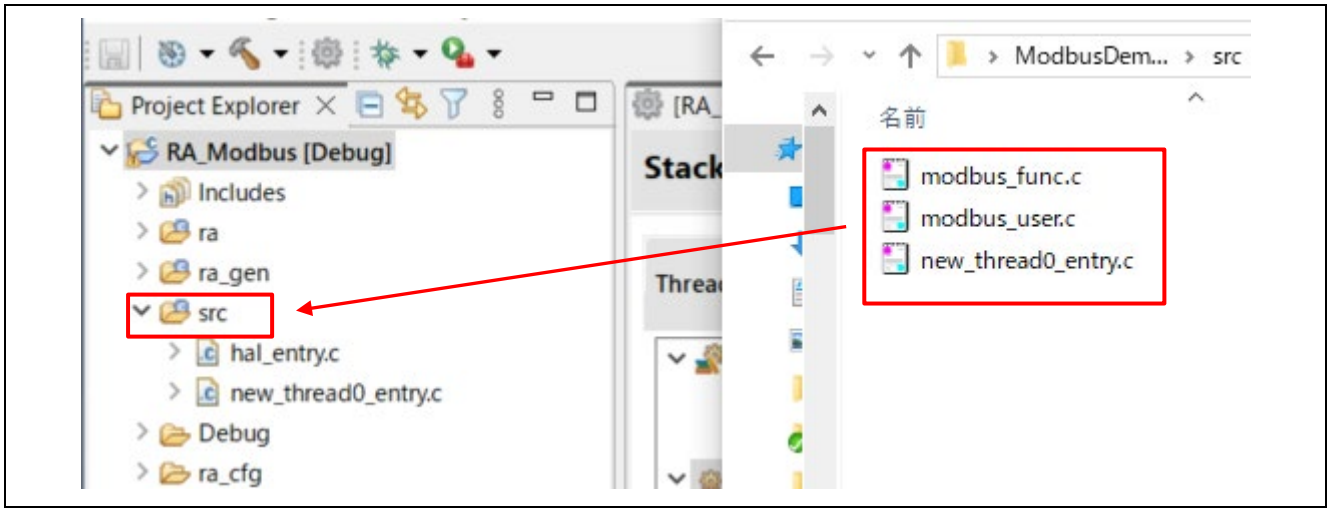
6. Set PHY Reset Terminals
   Change the "Mode" of “Pins” → ”Ports” → ”P7” → "P708" to "Output mode (Initial High)".

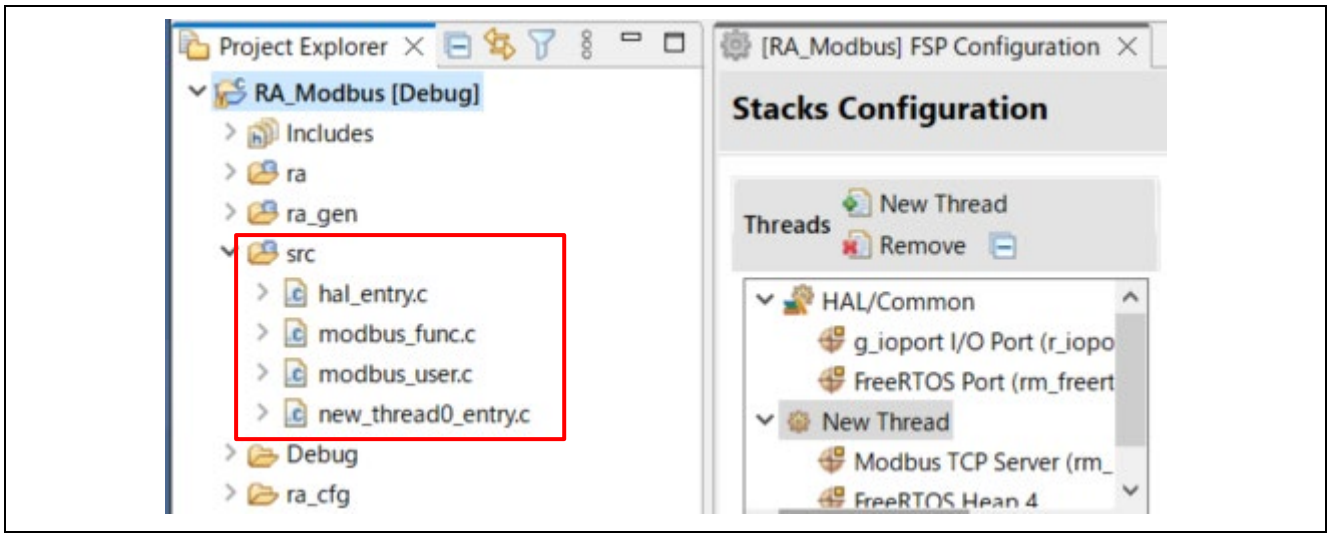


7. Generate the code
   Generate the code with "Generate Project Content".



8. Add Modbus Sample Application
   Copy modbus_func.c, modbus_user.c, and new_thread0_entry.c from the src folder of the sample program package to the src folder of the project and overwrite them.

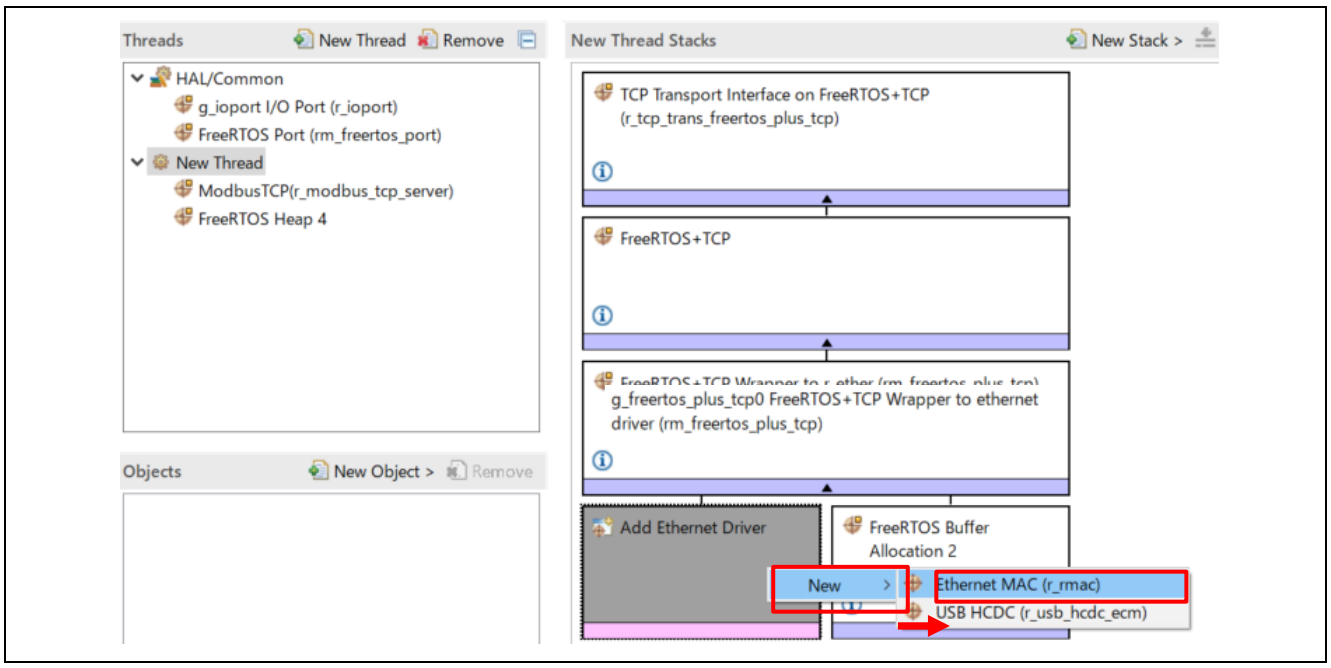




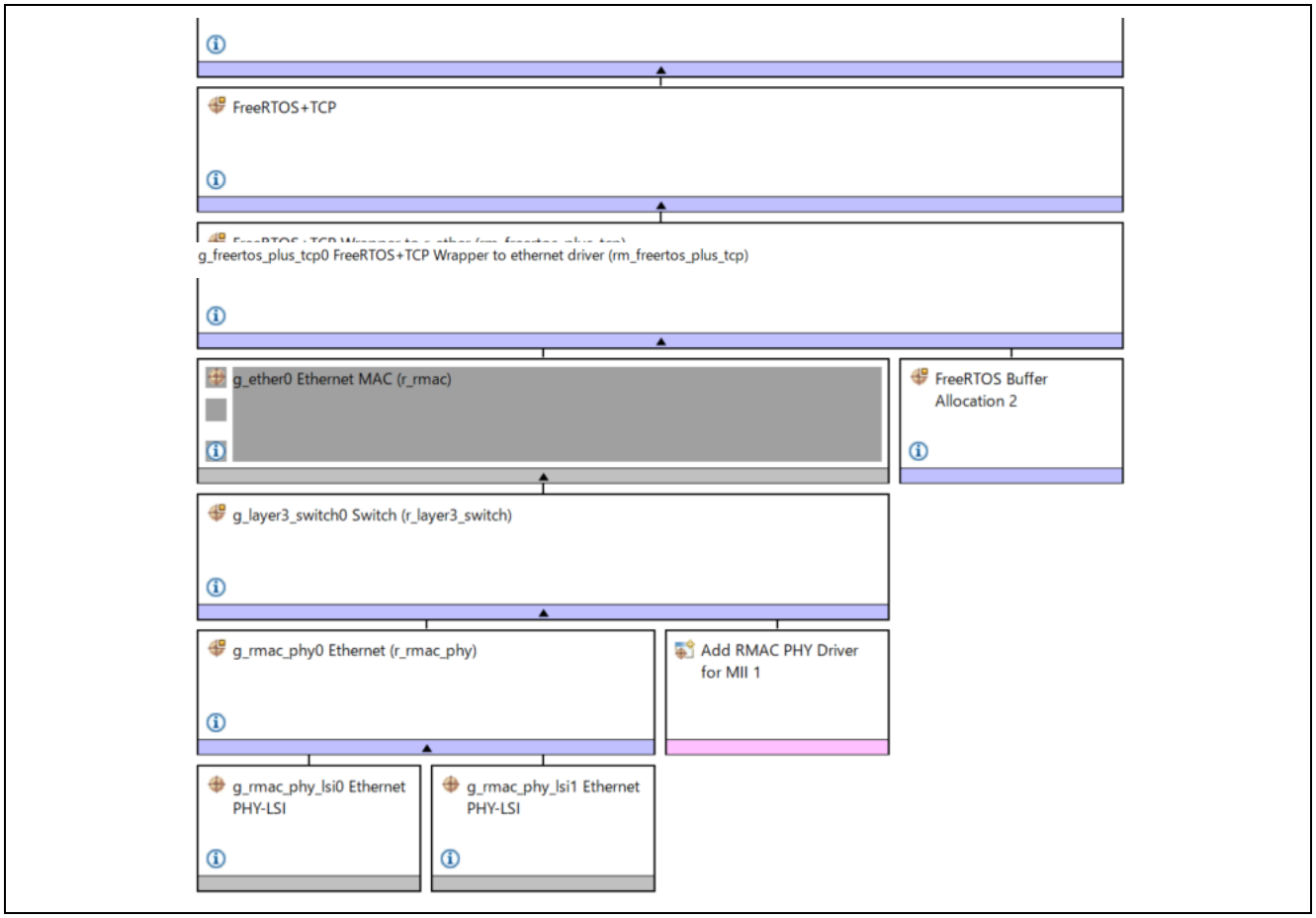
6.2.5 EK-RA8P1 Creating Procedures

This section describes the procedures for creating EK-RA8P1.

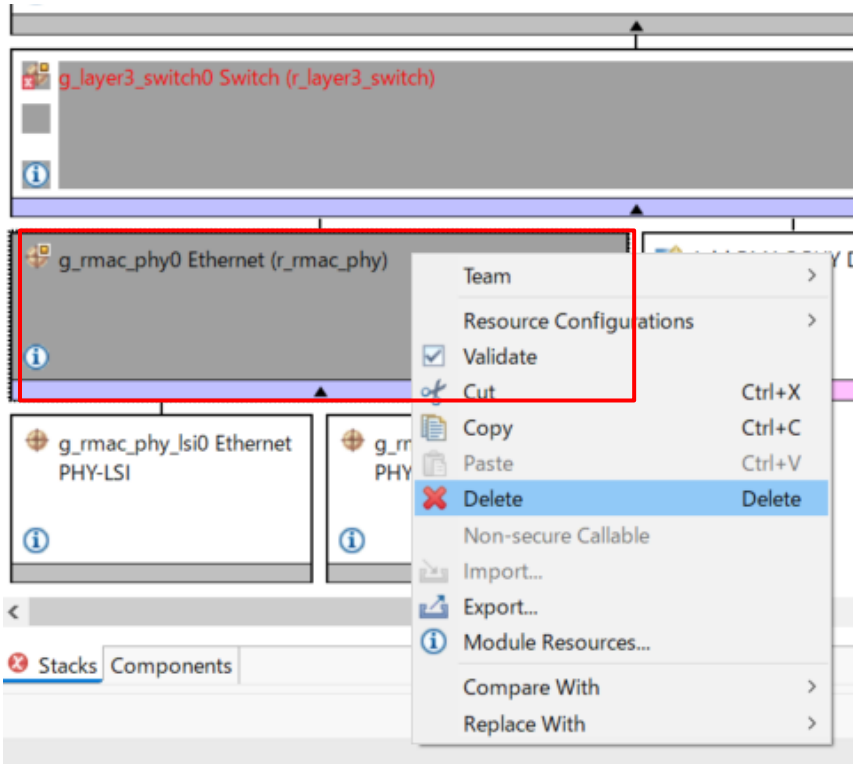
1. Add Ethernet Driver
   Click “New” → “Ethernet (r_rmac)” to “Add Ethernet Driver”.



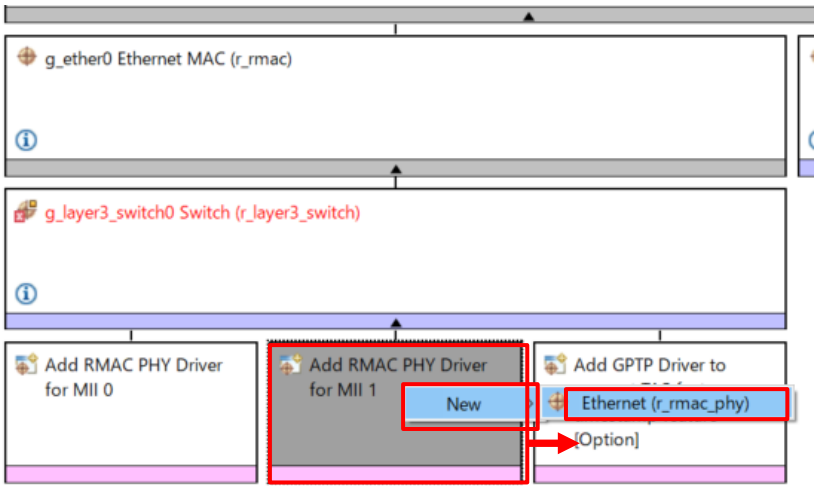
The stack will be configured as follows:



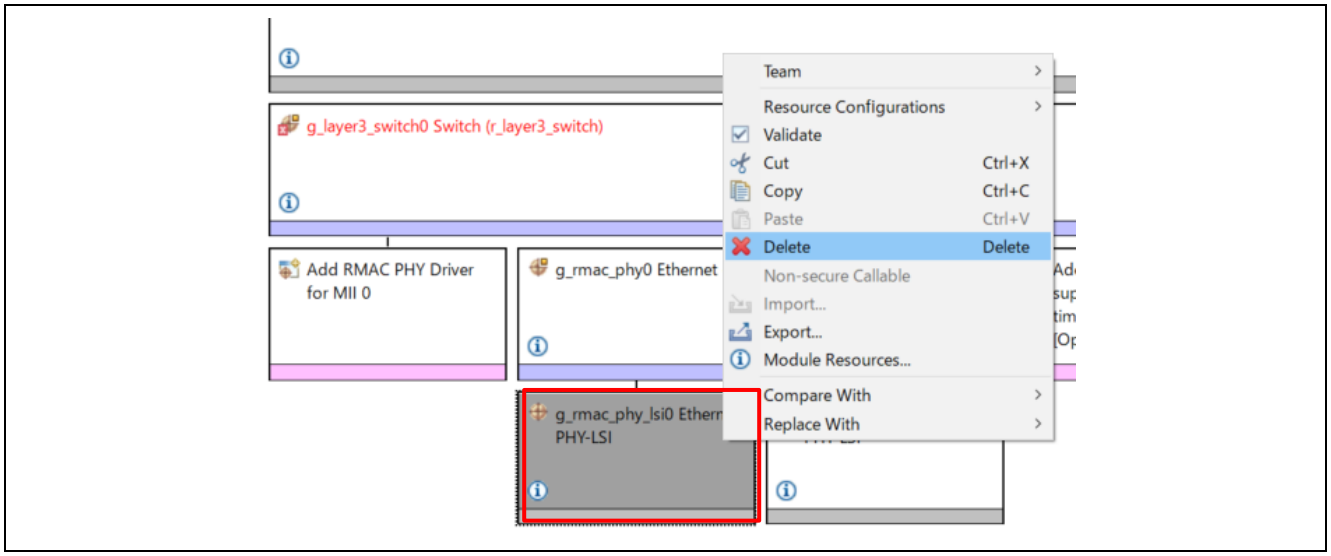
Right-click on “g_mac_phy0 Ethernet (r_rmac_phy)”, select Delete, and then click “OK” on the “Remove Stack Elements” pop-up.



Click “New” → “Ethernet (r_rmac_phy)” to “Add RMAC PHY Driver for MII1”.

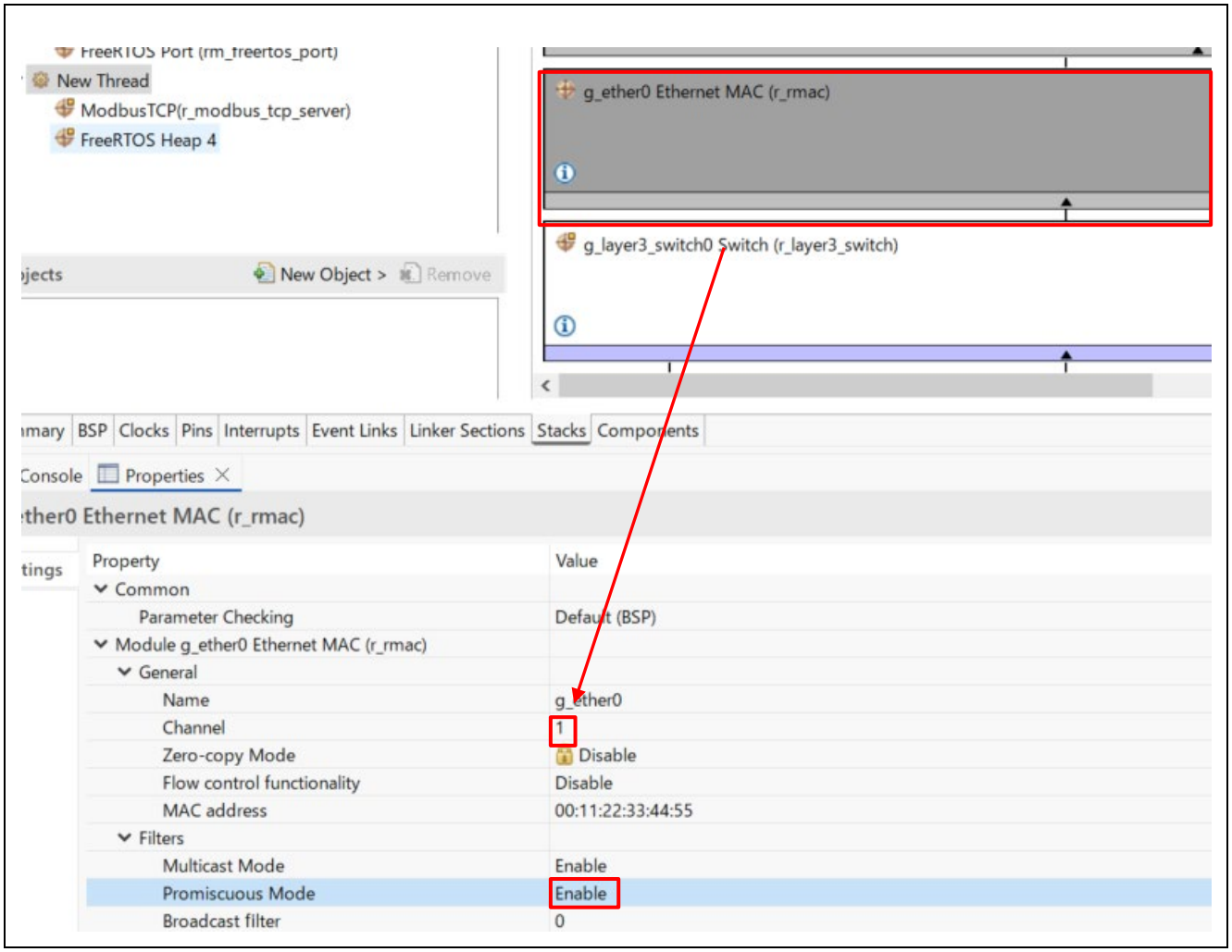


Right-click on “g_rmac_phy_lsi0 Ethernet PHY-LSI”, select Delete, and then click “OK” on the “Remove Stack Elements” pop-up.



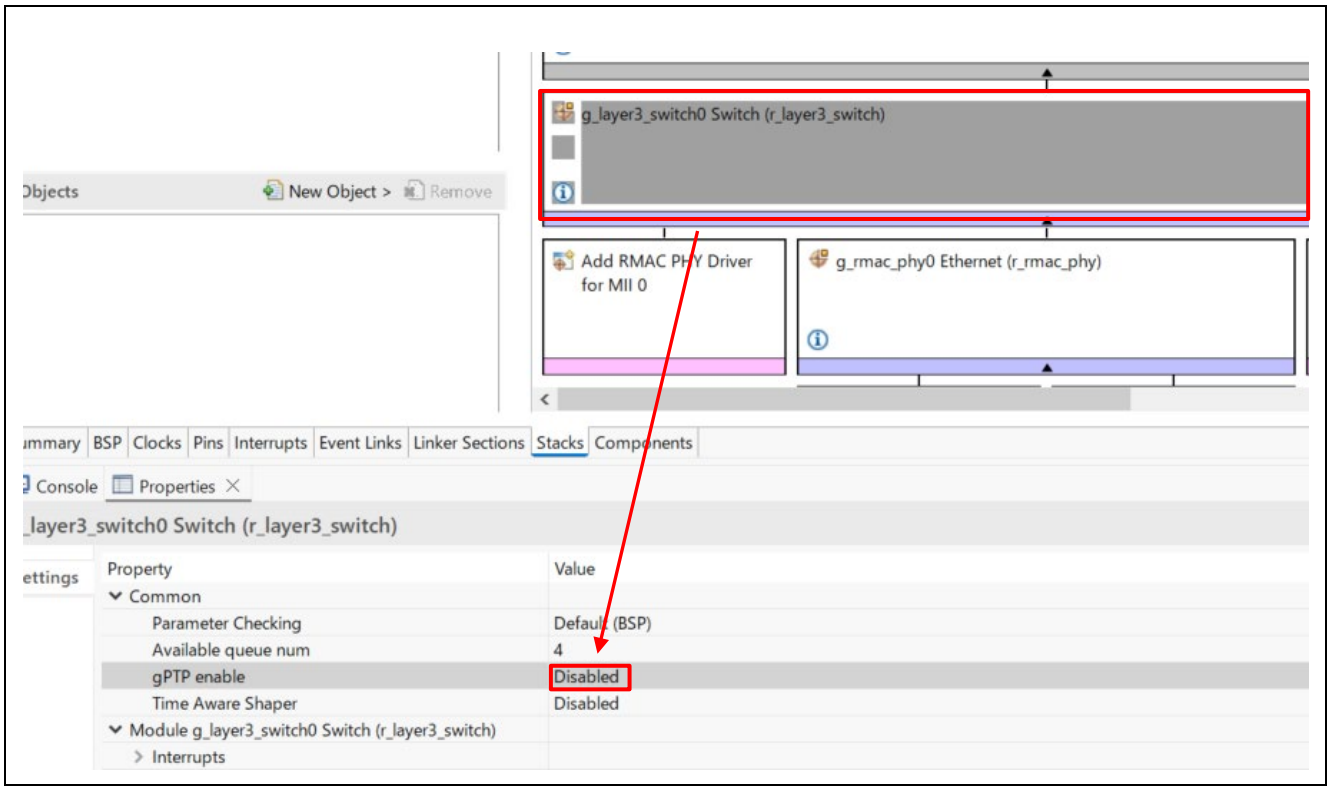
2. Set r_rmac
   Open “Properties” of “g_ether0 Ethernet MAC (r_rmac)” in “Stacks” and change “Channel” and “Promiscuous Mode” in “Module g_mac_phy0 Ethernet (r_rmac_phy)” to the following values.

   Channel：**1**
   Promiscuous Mode：**Enable**



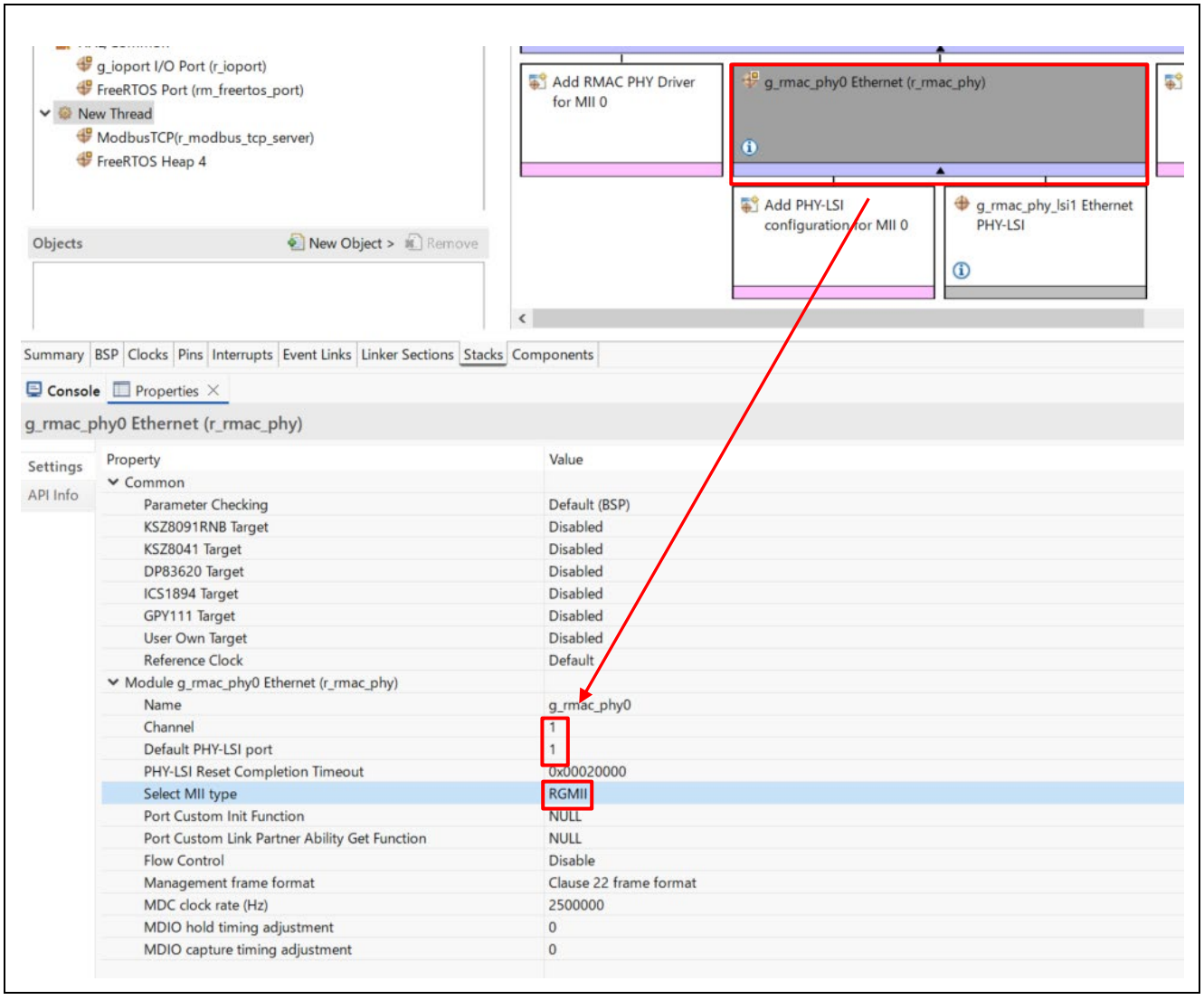
3. Set r_layer3_switch
   Open “Properties” of “g_layer3_switch0 Switch (r_layer3_switch)” in “Stacks” and change “gPTP enable” to the following values.

   gPTP enable：**Disabled**

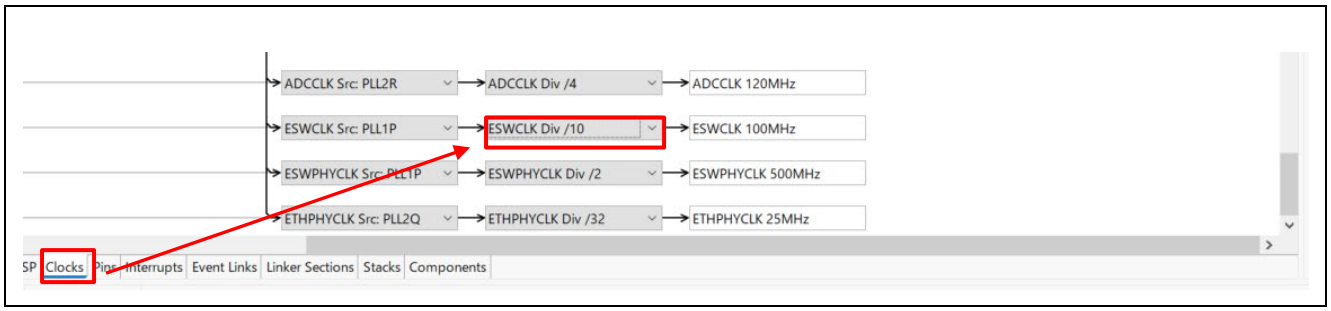


4. Set r_rmac_phy
   Open “Properties” of “g_mac_phy0 Ethernet (r_rmac_phy)” in “Stacks” and change “Channel”, “Default PHY-LSI port” and “Select MII type” in “Module g_mac_phy0 Ethernet (r_rmac_phy)” to the following values.

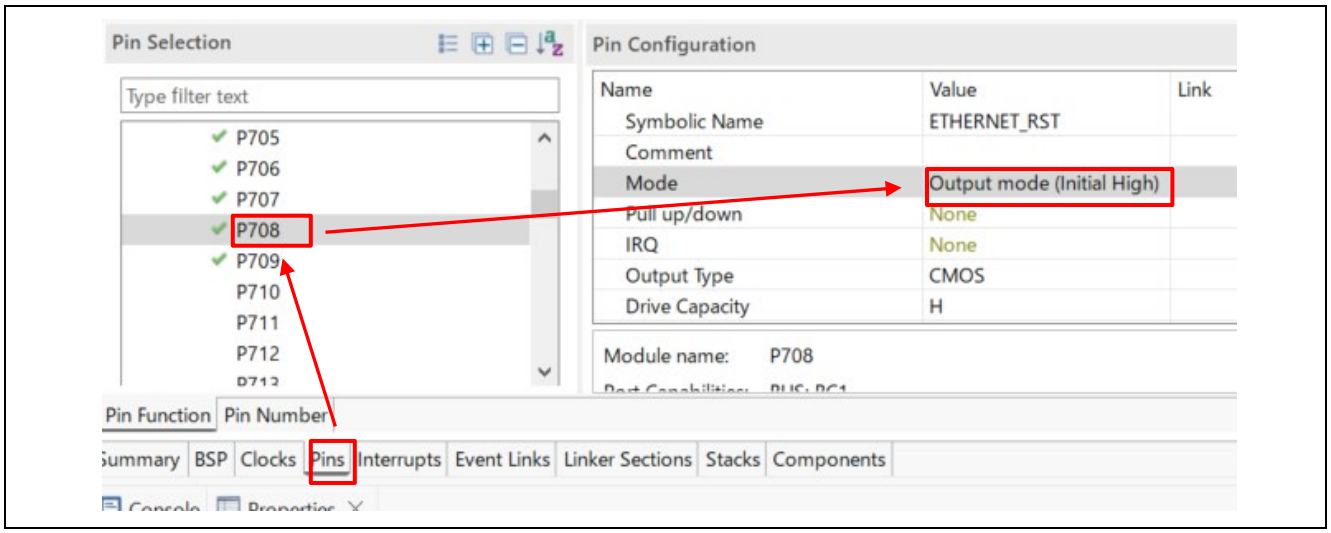
   Channel：**1**
   Default PHY-LSI port：**1**
   Select MII type：**RGMII**



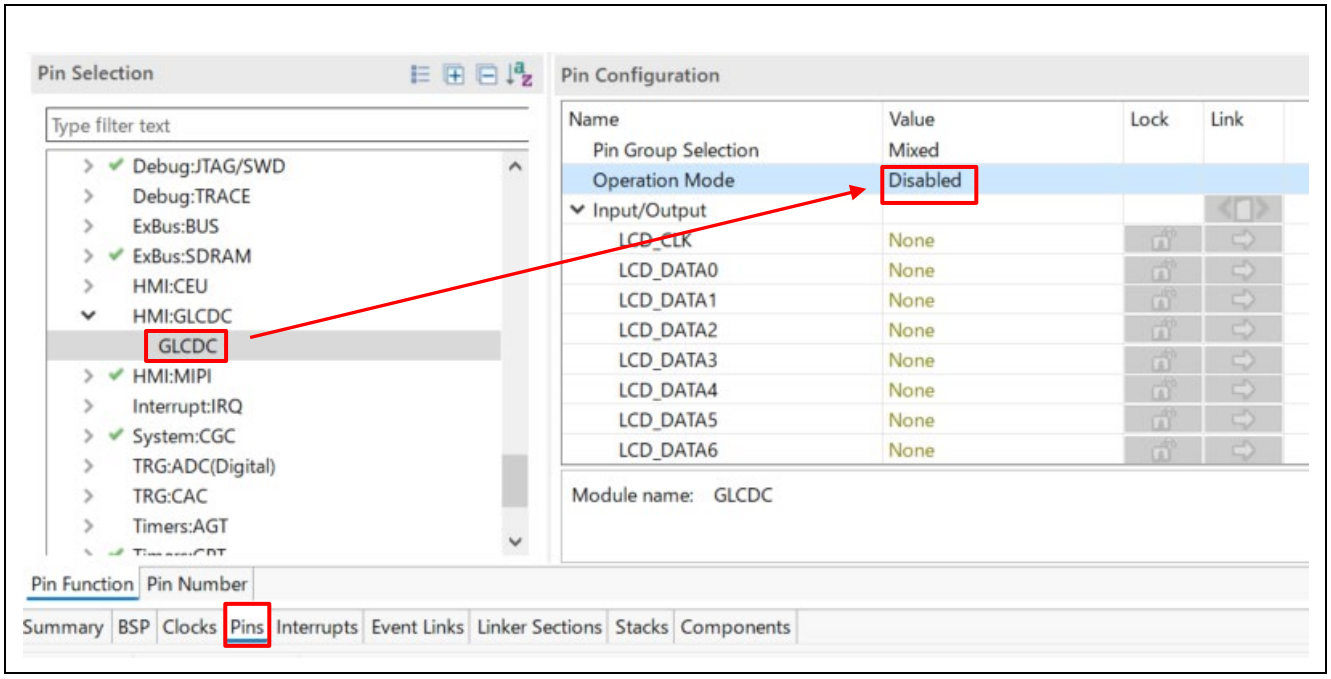
5. Set ESW Clock
   Change the “ESWCLK Div” at the bottom of “Clocks” to “ESWCLK Div /10”.



6. Set PHY Reset Terminals
   Change the "Mode" of “Pins” → ”Ports” → ”P7” → "P708" to "Output mode (Initial High)".

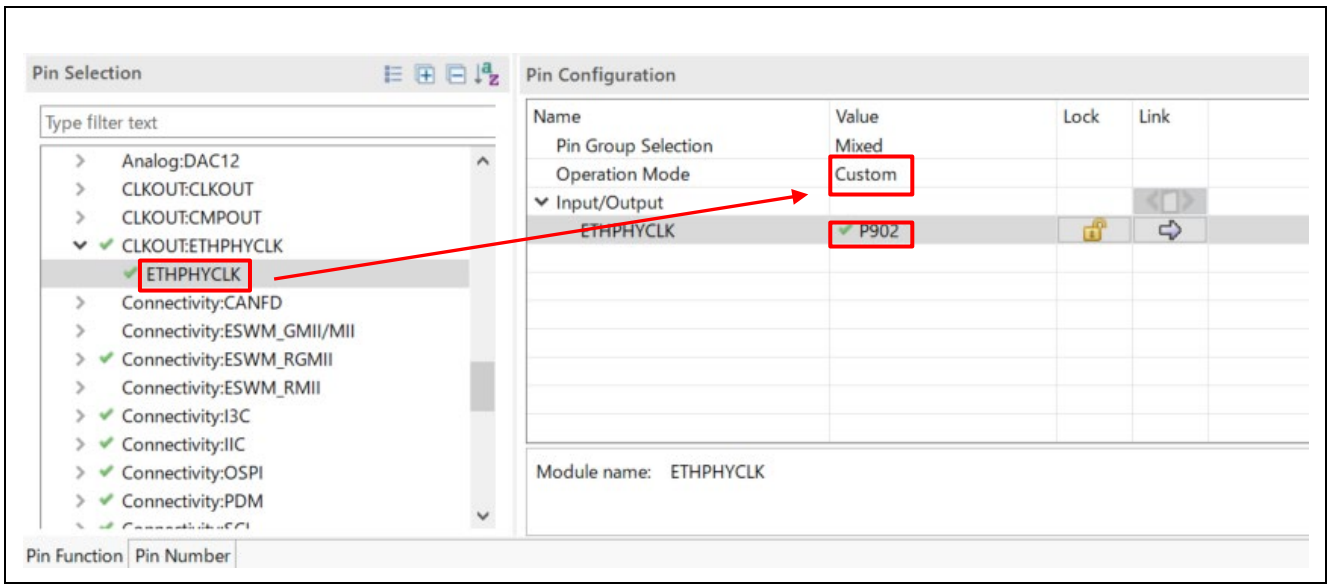


7. Set PHY Clock Terminals
   Change the "Operation Mode" of “Pins” → ”Peripherals” → ” HMI: GLCDC” → " GLCDC " to "Disabled".

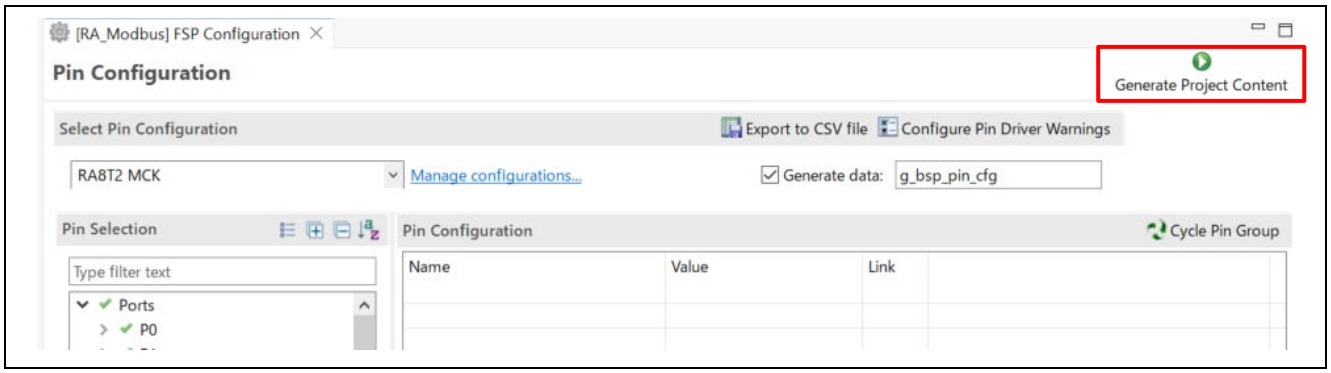


After changing the "Operation Mode" of "Pins" → "Peripherals" → " CLKOUT: ETHPHYCLK " → " ETHPHYCLK " to "Custom", change the pins as follows:

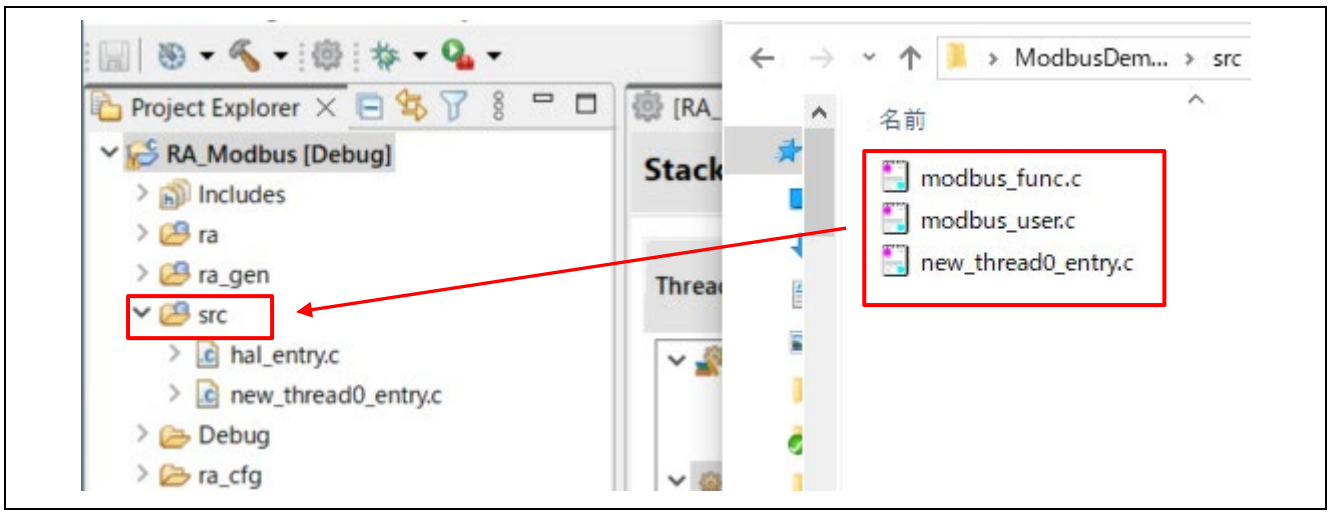
ETHPHYCLK：**P902**

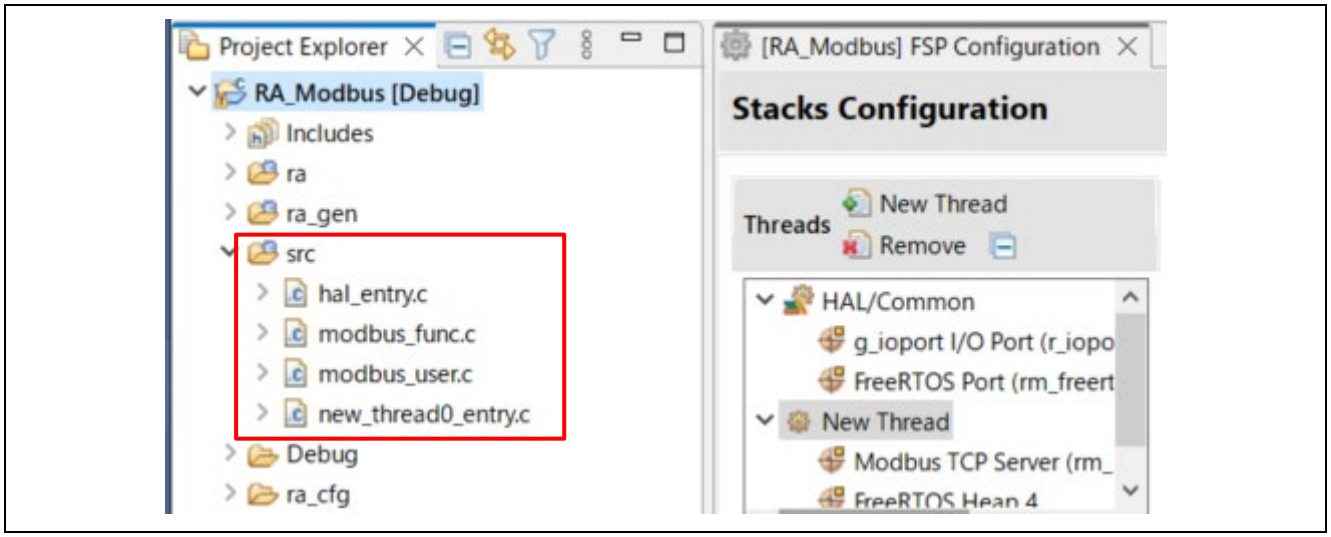


8. Generate the code
   Generate the code with "Generate Project Content".



9. Add Modbus Sample Application
   Copy modbus_func.c, modbus_user.c, and new_thread0_entry.c from the src folder of the sample program package to the src folder of the project and overwrite them.

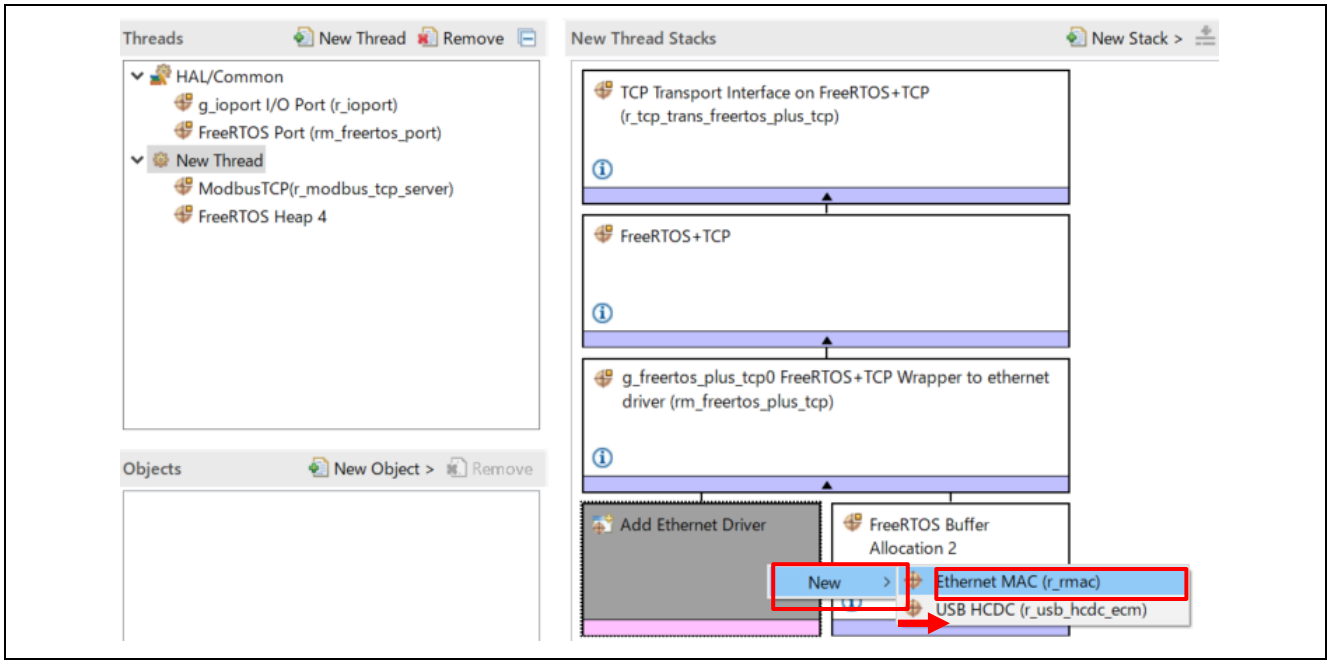




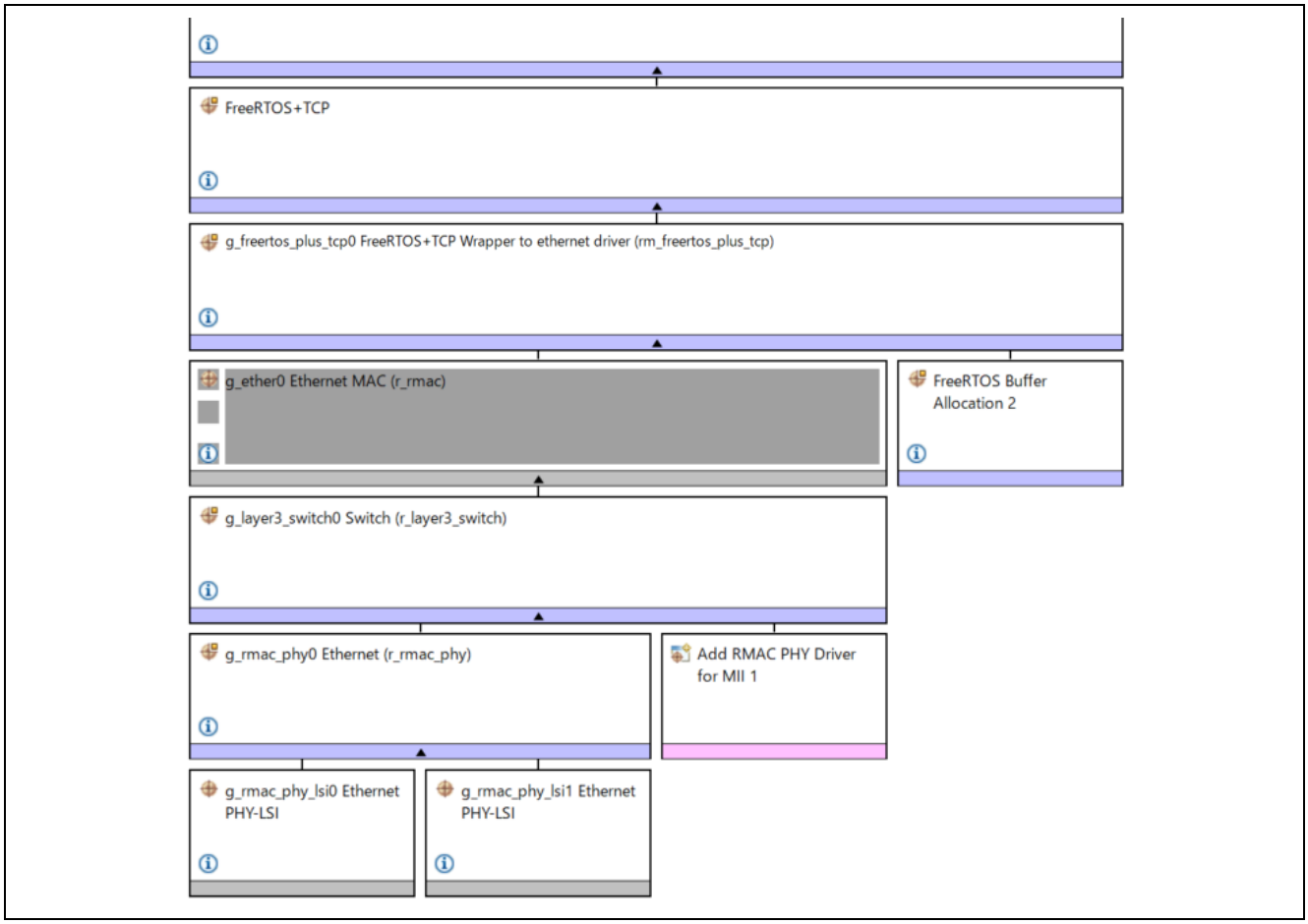
6.2.6 EK-RA8M2 Creating Procedures

This section describes the procedures for creating EK-RA8M2.

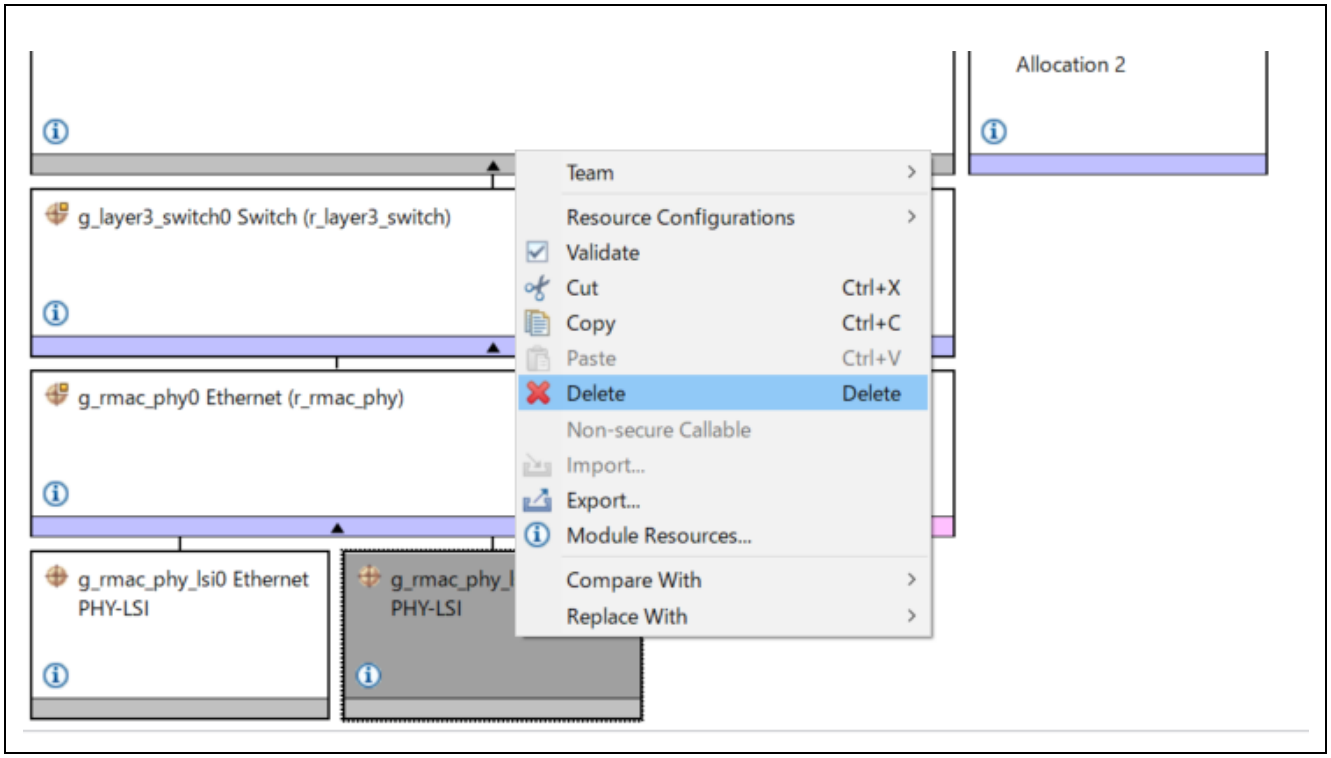
1. Add Ethernet Driver
   Click “New” → “Ethernet (r_rmac)” to “Add Ethernet Driver”.



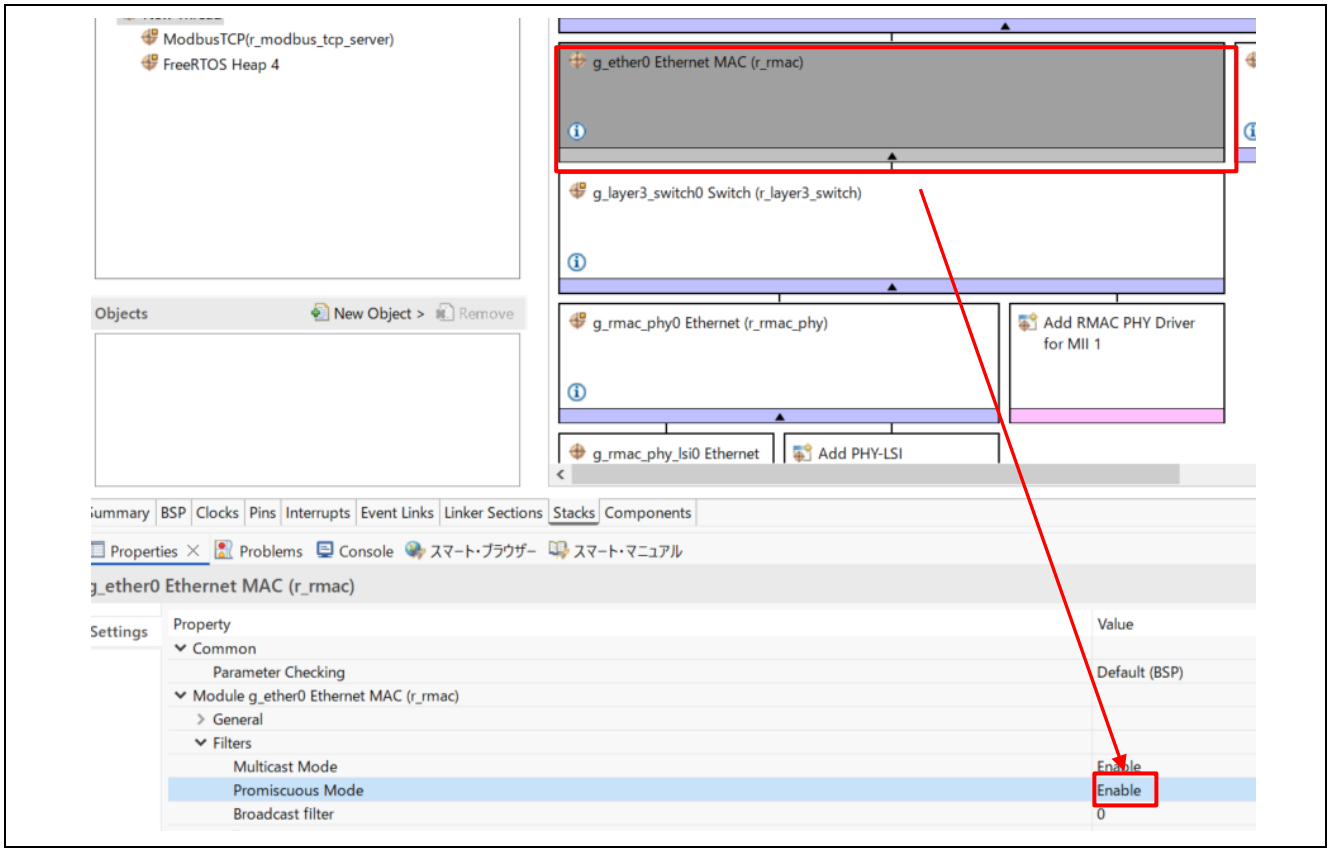
The stack will be configured as follows:



Right-click on “g_rmac_phy_lsi1 Ethernet PHY-LSI”, select Delete, and then click “OK” on the “Remove Stack Elements” pop-up.

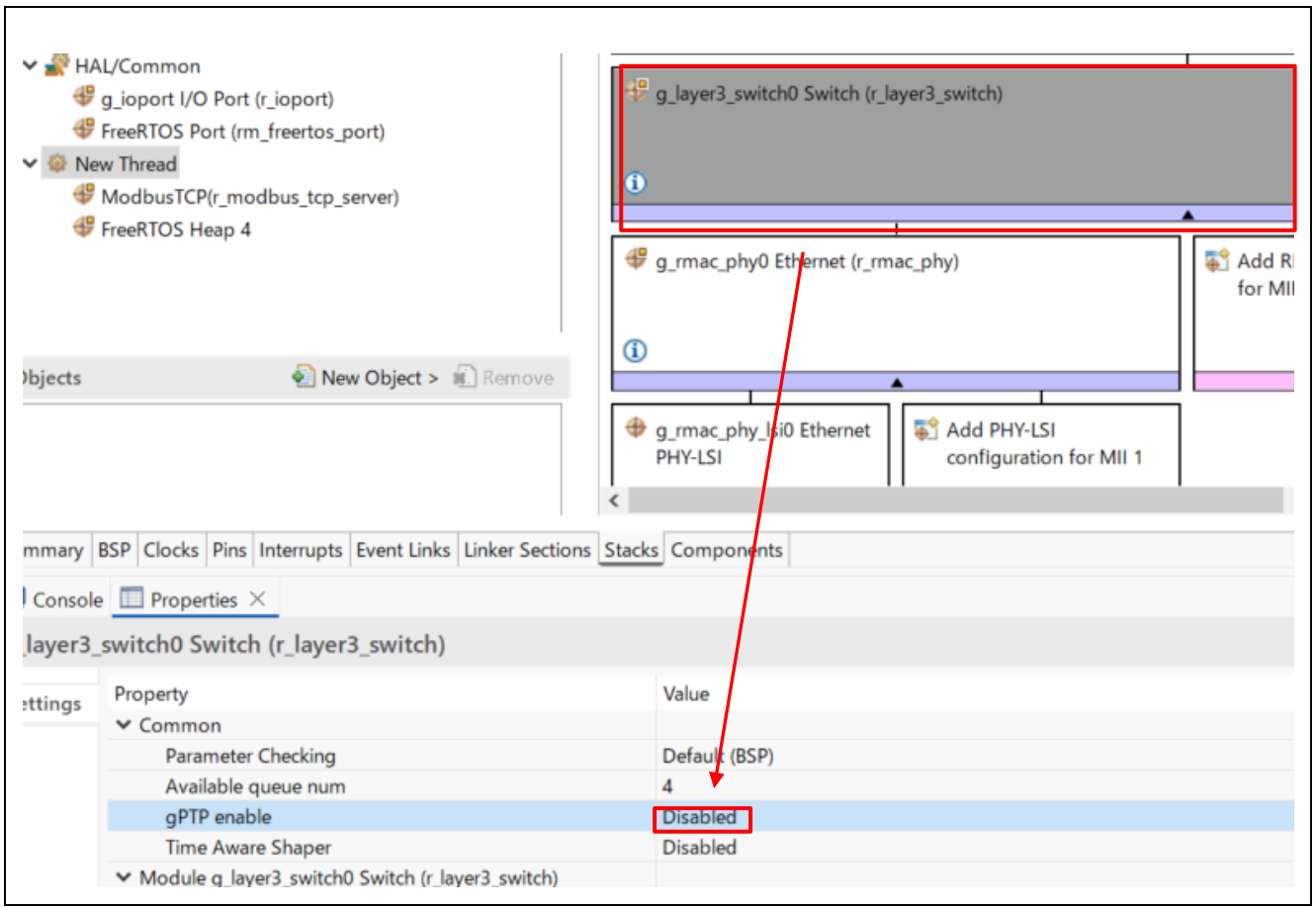


2. Set r_rmac
Change the “Stacks” → ”g_ether0 Ethernet MAC (r_rmac)” → ”Module g_ether0 Ethernet MAC (r_rmac)” → “Promiscuous Mode” in “Filters” to “Enable”.



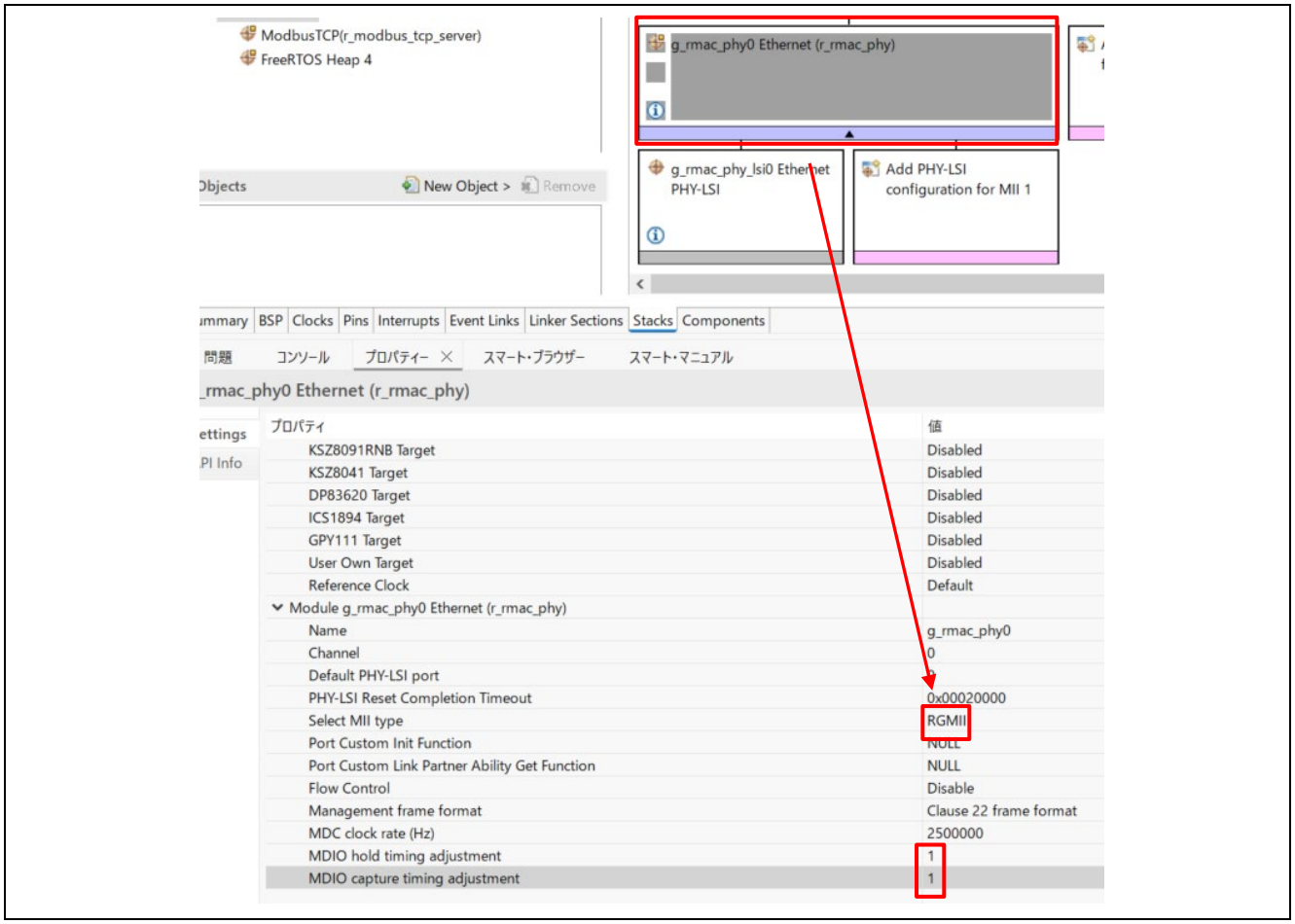
3. Set r_layer3_switch
   Open “Properties” of “g_layer3_switch0 Switch (r_layer3_switch)” in “Stacks” and change “gPTP enable” to the following values.

   gPTP enable：**Disabled**

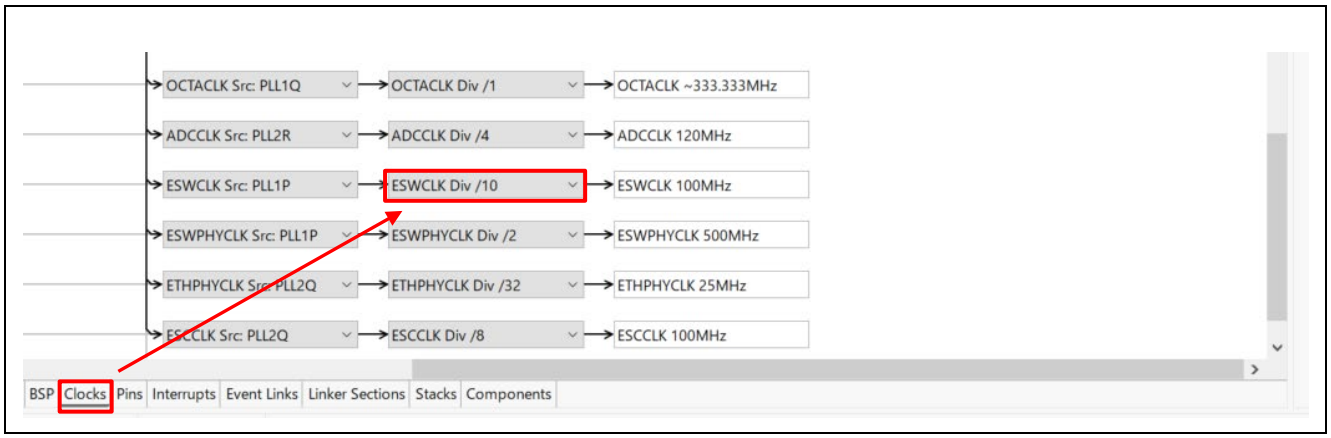


4. Set r_rmac_phy
Open “Properties” of “g_mac_phy0 Ethernet (r_rmac_phy)” in “Stacks” and change “Select MII type”, ” MDIO hold timing adjustment” and “MDIO capture timing adjustment” in “Module g_mac_phy0 Ethernet (r_rmac_phy)” to the following values.

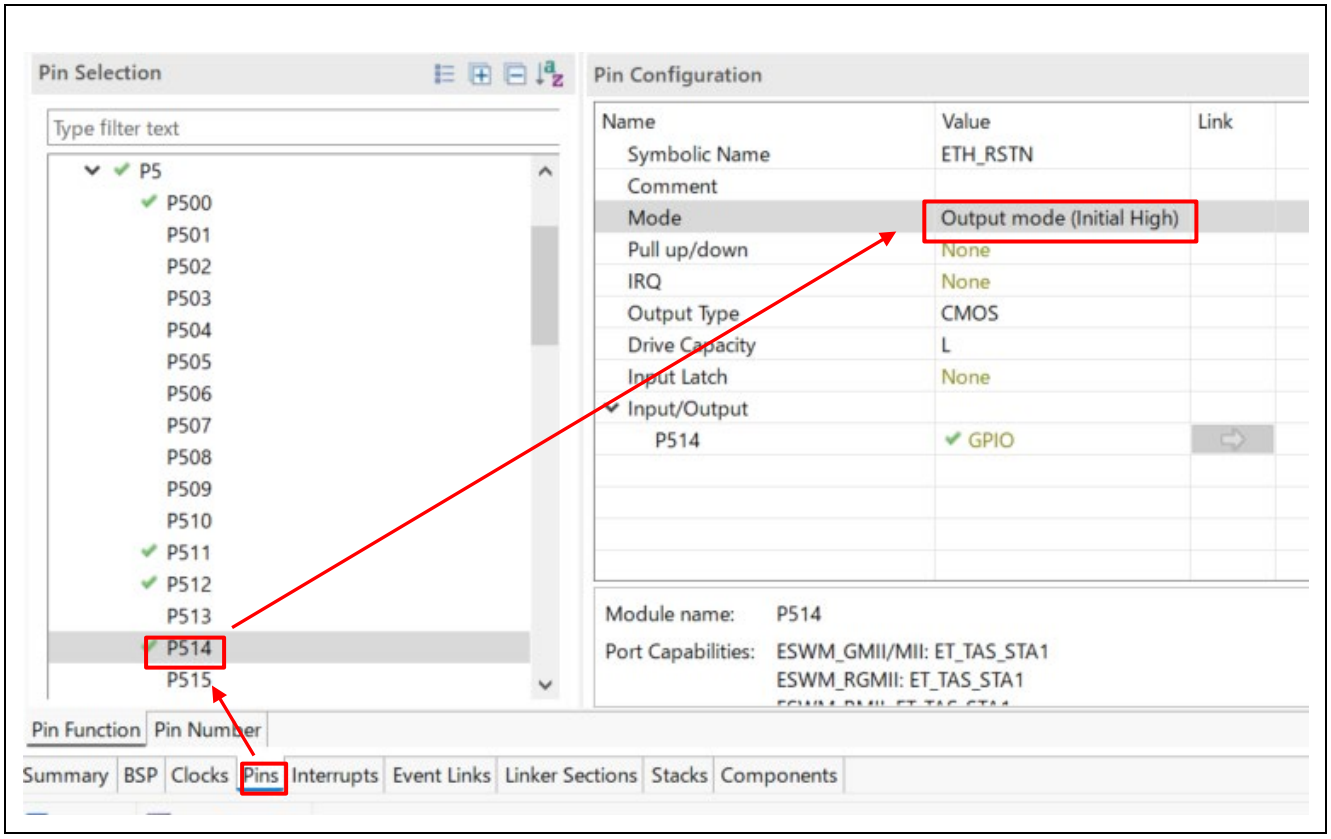
Select MII type：**RGMII**
MDIO hold timing adjustment：**1**
MDIO capture timing adjustment：**1**



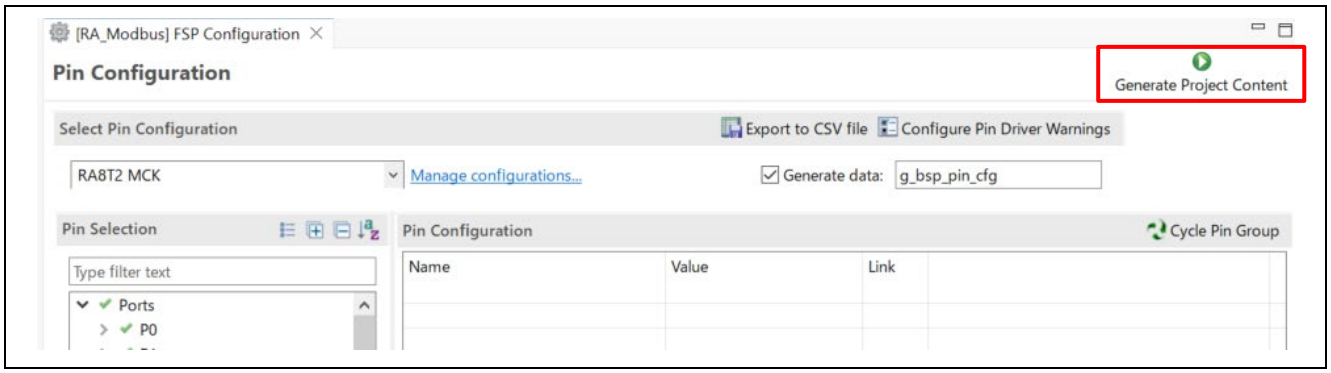
5. Set ESW Clock
Change the “ESWCLK Div” at the bottom of “Clocks” to “ESWCLK Div /10”.



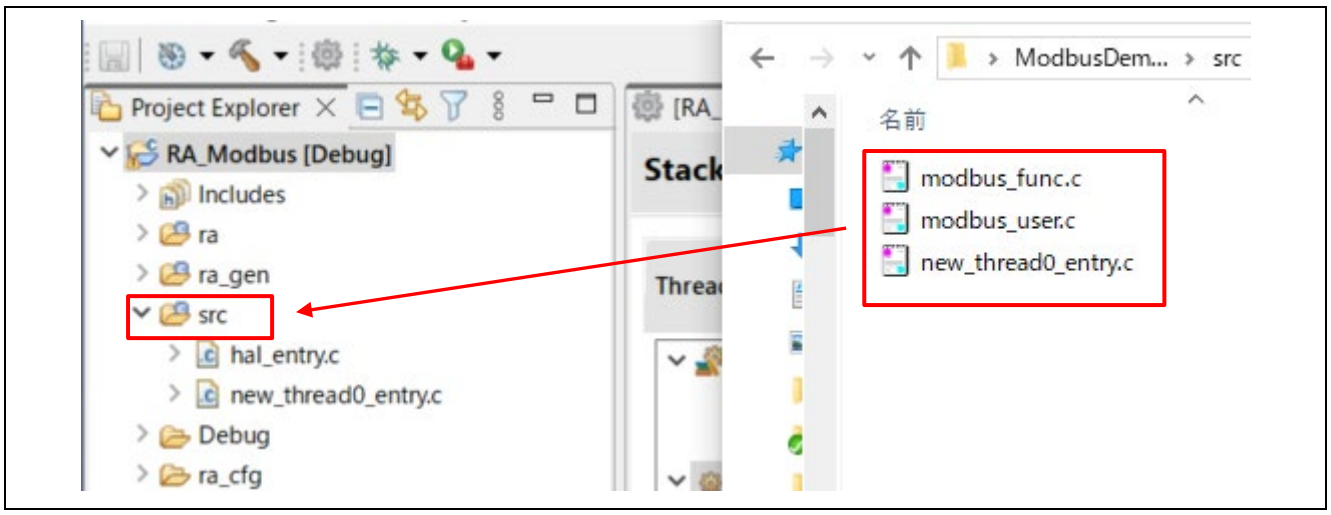
6. Set PHY Reset Terminals
   Change the "Mode" of “Pins” → ”Ports” → ”P5” → "P514" to "Output mode (Initial High)".

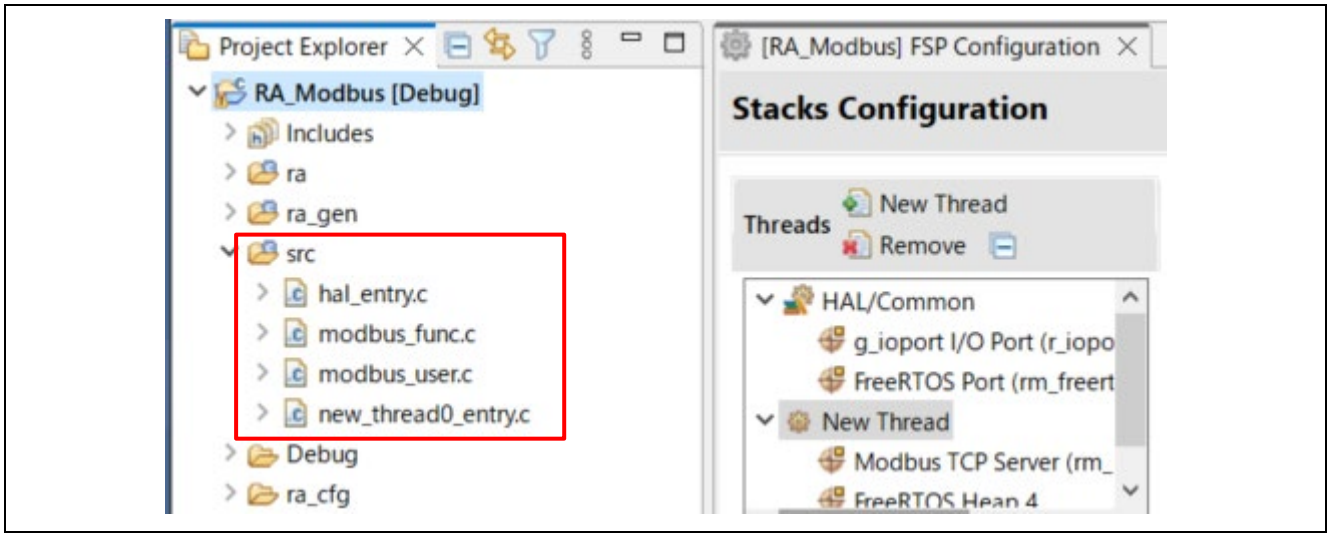


7. Generate the code
   Generate the code with "Generate Project Content".



8. Add Modbus Sample Application
   Copy modbus_func.c, modbus_user.c, and new_thread0_entry.c from the src folder of the sample program package to the src folder of the project and overwrite them.

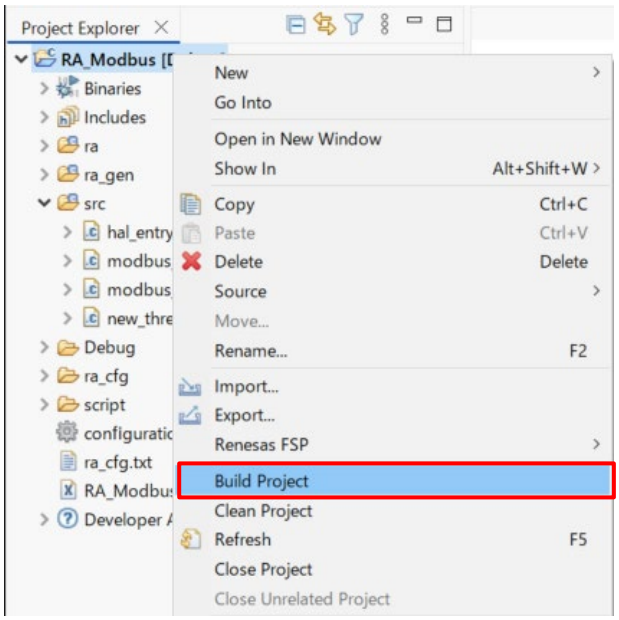




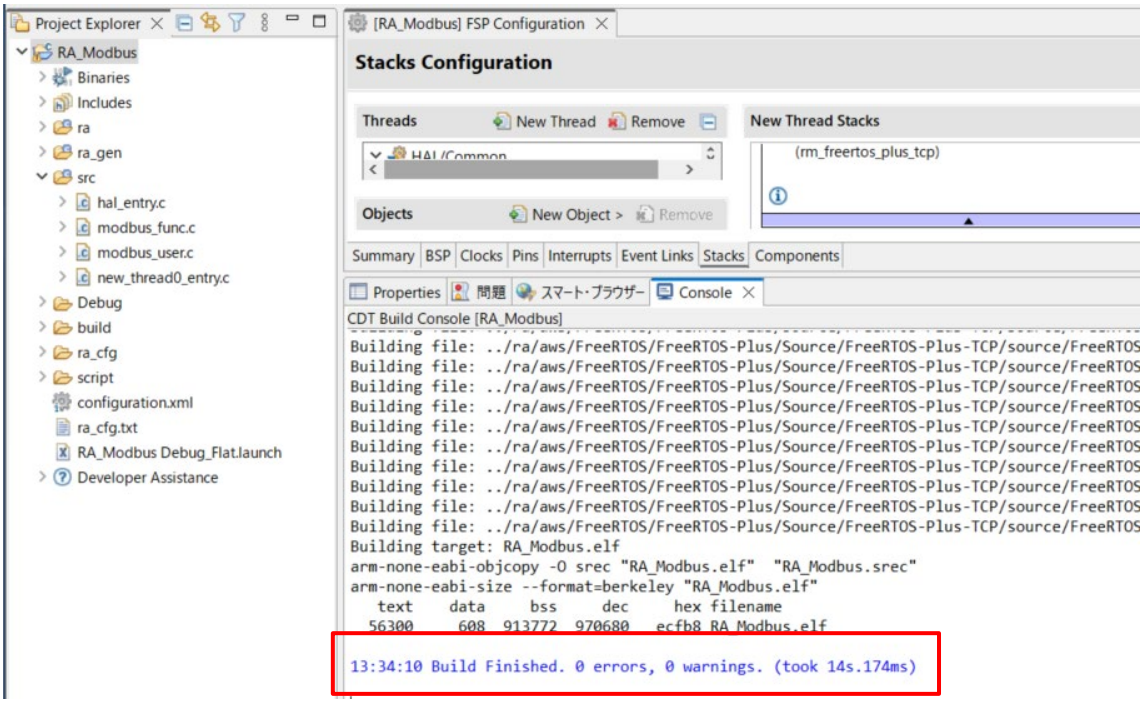
7. Execution of Modbus Sample Project

Before you begin, read "**5. Evaluation Board Connection Setup**" to complete the hardware connections. Also, read "**6. Setting Up the Modbus Sample Project**" to complete the preparation of the Modbus sample project.

1. Build the project
   Right-click on the project from the “Project Explorer”, then select “Build Project”.

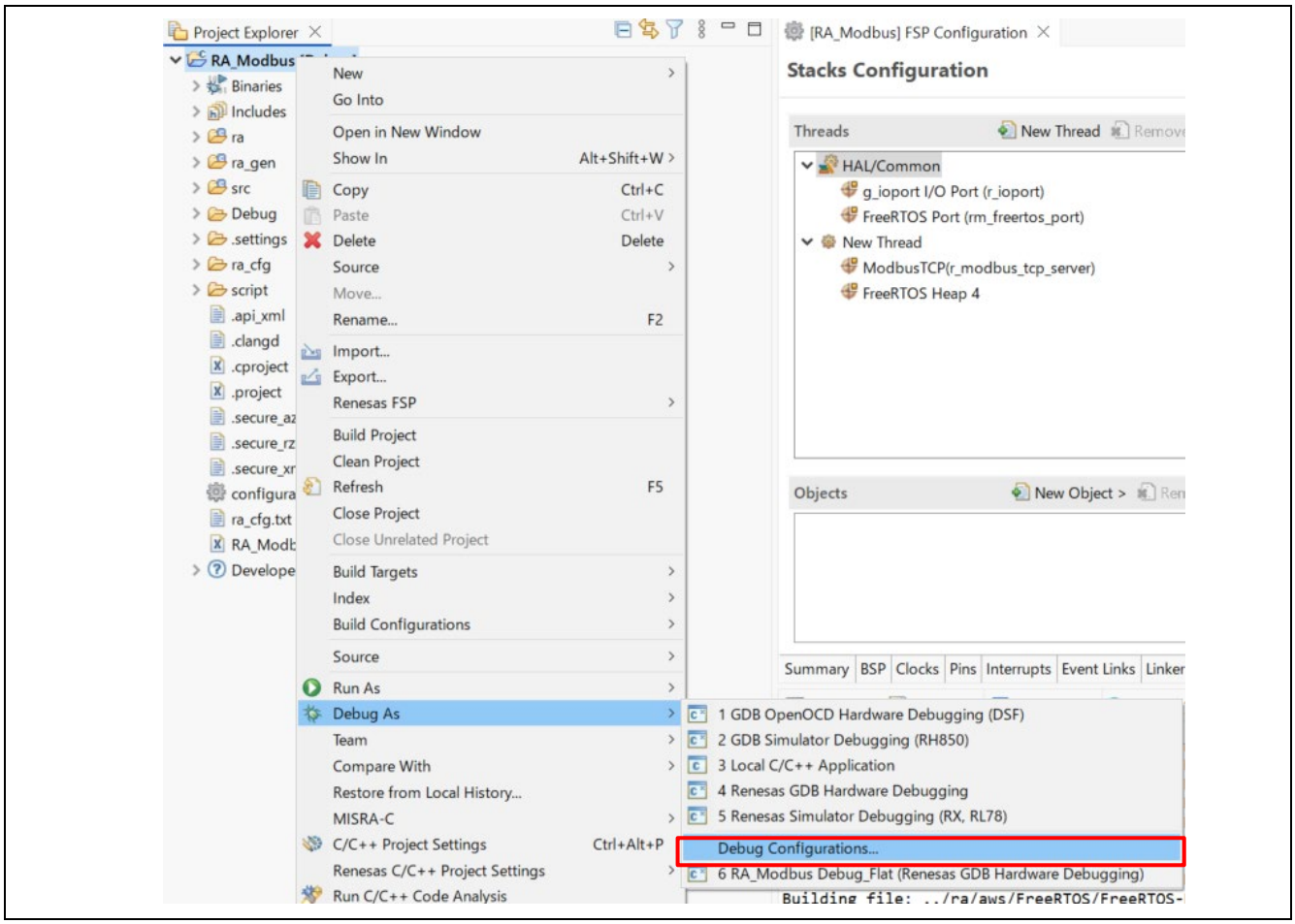


At this time, make sure there are no build errors.

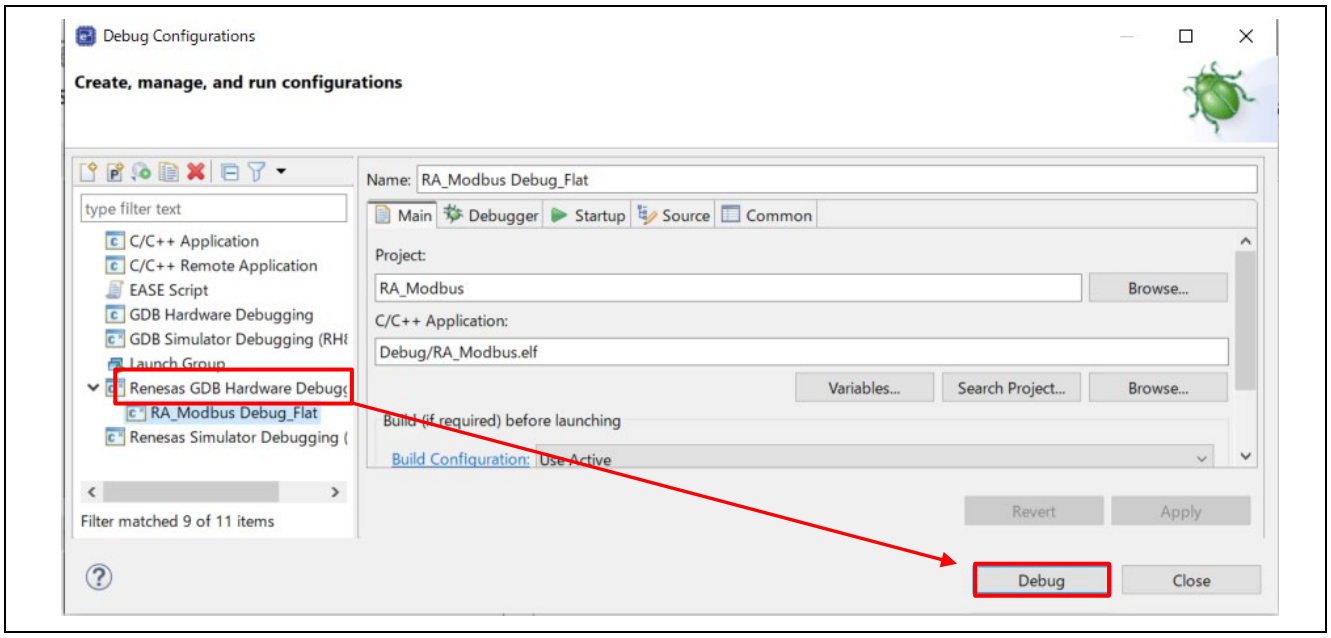


2. Download the application and run the debugger.
   To start debugging, follow the steps below:

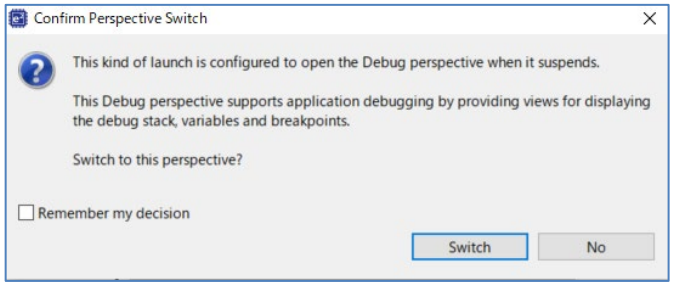
   In “Project Explorer” view, right-click on the project to be debugged and select “Debug As” → “Debug Configurations”.



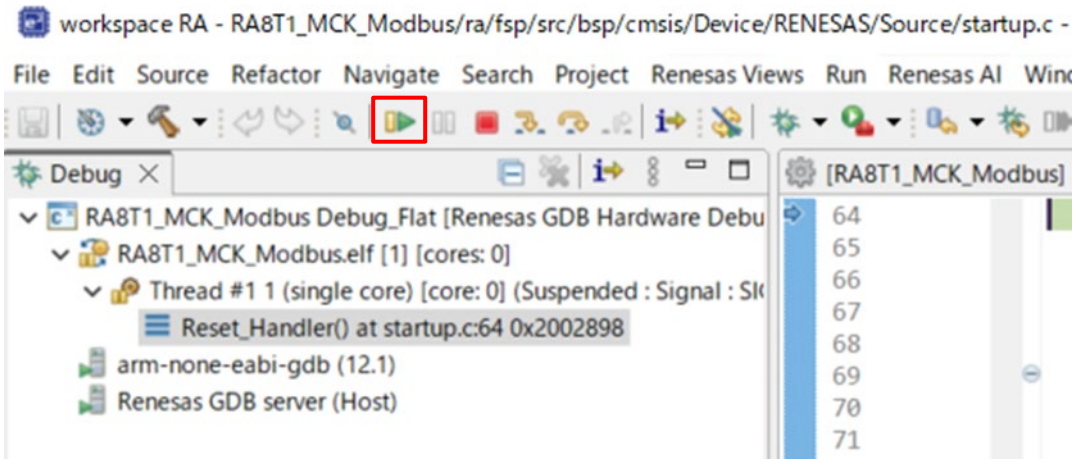
3. Program download.
   Select “Renesas GDB Hardware Debugging” → “xxxxx Debug_Flat”, then click “Debug”.



The following dialog appears. Switch to the debug screen.



4. Program starts.
   Click the **Resume** button.
   When debugging starts, the program is suspended at main.c.
   Click the **Resume** button again to run the program.



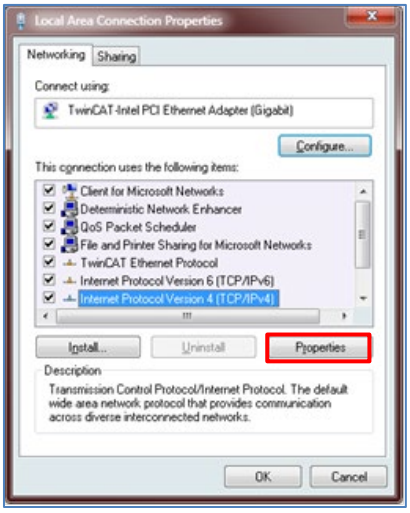
8. Modbus Communication Using Modbus Demo Application

This section demonstrates the procedure for checking the demo operation of the Modbus sample application using the Modbus demo application. For information on configuring the Modbus protocol stack, see "**9.1. Appendix A: Modbus Protocol Stack Configuration**".

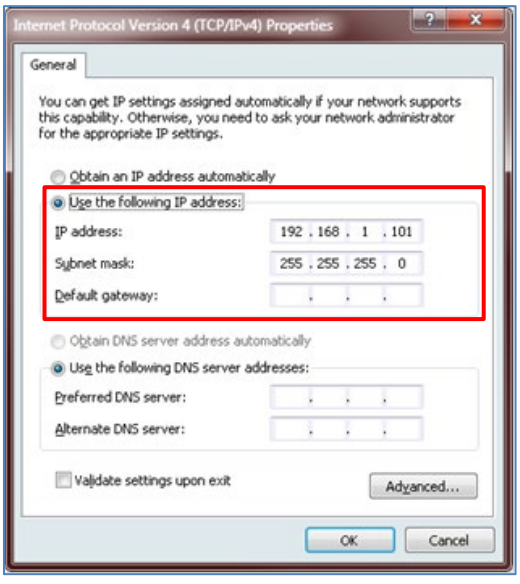
8.1 IP Address Setting

To run the Modbus sample application, it is necessary to set the IP address of the PC running the evaluation tool, which is the client, to the same domain as the evaluation board.

1. Open the network connections list.
   Control panel → **Network and Sharing Center** → **Change adapter settings.**
   Double-click (or right-click) on the Local Area Connection, then select **Properties**.
   Select TCP/IPv4 and click the **Properties** button.

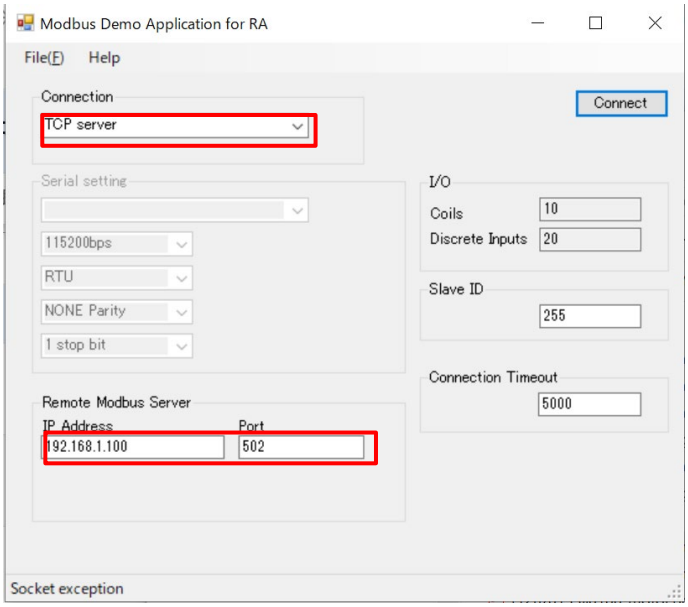


2. Set IP-address and subnet mask
   IP Address : 192.168.1.101, subnet mask : 255.255.255.0.



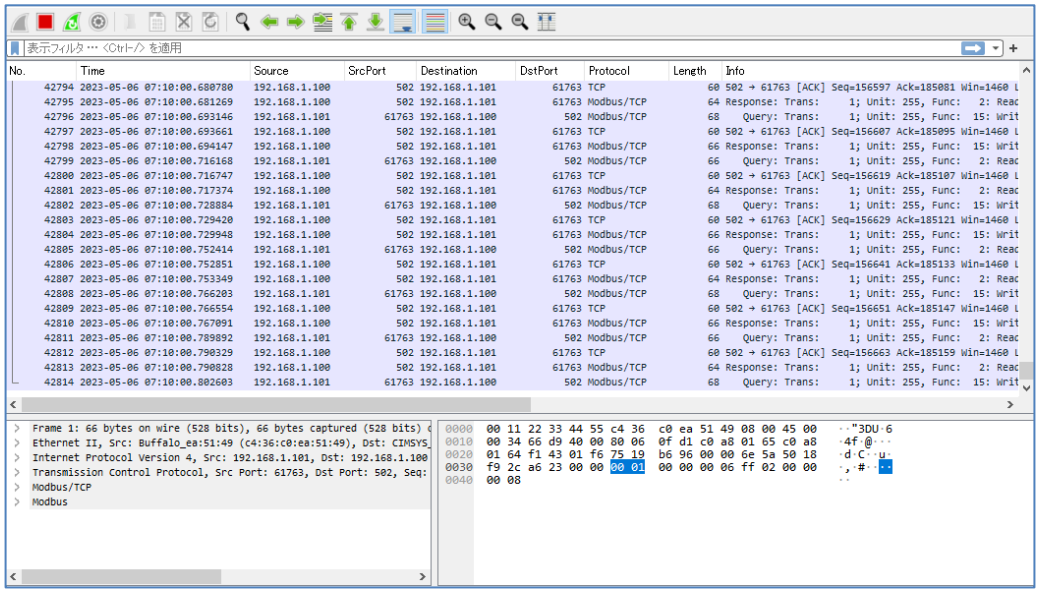
8.2 Setting up of Modbus Demo Application

1. Start "ModbusDemoApplication.exe" included in this sample program package.
   Under "Connection", select TCP Server and set the server IP address (for example, "192.168.1.100") and port number (for example, "502").



2. Click the **Connect** button. Once connected, the LED on the evaluation board should start blinking, indicating active Modbus communication.

3. Check the Modbus communication status by using a packet analysis tool like Wireshark.

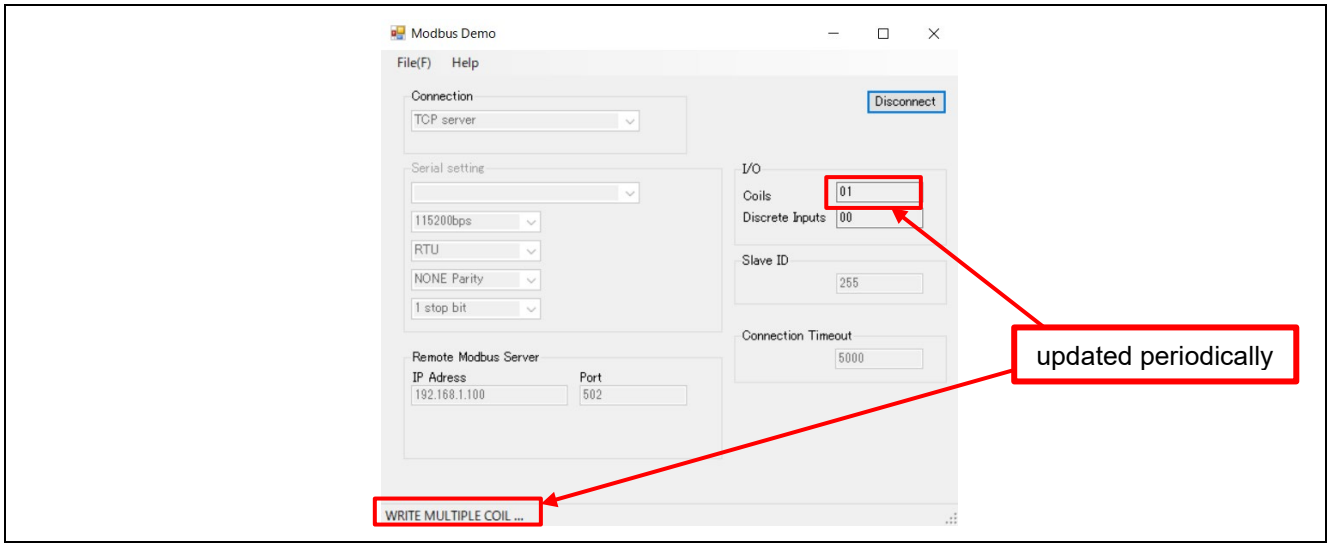


8.2.1 Modbus Demo Application Specification

The LED blinking can be controlled dynamically by communicating with the PC through the Modbus TCP protocol.

For this control, Read Coil and Write Coil function codes are used.

After clicking the "Connect" button, LEDs 1 to 3 will blink periodically.



9 Appendix

9.1 Appendix A: Modbus Protocol Stack Configuration

The Modbus protocol stack configuration is described below.

Configuration	Options	Default	Description
Common			
Parameter Checking	・Default (BSP) ・Enable ・Disable	Default (BSP)	If the selected code for parameter checking is included in the build.
Accept Task Stack Size	Legal values are 0x400 or higher.	0x400	Accept task stack size. Legal values are 0x400 or higher.
Accept Task Priority	Legal values range from 0 through (Max Priorities - 1)	2	Accept task priority. Legal values range from 0 through (Max Priorities - 1). Also, set Accept Task, Receive Task, and Service Task with the same priority.
Receive Task Stack Size	Legal values are 0x400 or higher.	0x400	Receive task stack size. Legal values are 0x400 or higher.
Receive Task Priority	Legal values range from 0 through (Max Priorities - 1)	2	Receive task priority. Legal values range from 0 through (Max Priorities - 1). Also, set Accept Task, Receive Task, and Service Task with the same priority.
Service Task Stack Size	Legal values are 0x400 or higher.	0x400	Service task stack size. Legal values are 0x400 or higher.
Service Task Priority	Legal values range from 0 through (Max Priorities - 1)	2	Service task priority. Legal values range from 0 through (Max Priorities - 1). Also, set Accept Task, Receive Task, and Service Task with the same priority.
Maximum Number of Clients	Legal values are from 1 to 3.	3	Maximum number of connection sockets for clients. Valid values are 1 to 3.
Receive Queue Length	Legal values are 8 or higher.	8	The length of the queue that passes data between tasks. Legal values are 8 or higher.
Server ID	Legal values are from 1 to 255.	255	Modbus server ID. Legal values are from 1 to 255.

Configuration	Options	Default	Description
Module Modbus TCP(r_modbus_tcp_server)			
Name	Name Must Be a Valid C Symbol	g_modbus_tcp_server0	Module name.
Callback for Function Code	Name Must Be a Valid C Symbol	function_code_callback	Enter the user callback function name for “Function Code”
Additional Port Number	Legal values range from 0 to 65535. Also, do not specify the Modbus TCP Server default port number “502” with this property.	0	Modbus TCP Server listens on both the default port “502” and the port number specified in this property. Enter 0 if you do not want to use a port other than the default. Also, do not specify the Modbus TCP Server default port number “502” with this property.
IP List Status	・Disable ・Enable	Enable	IP list enabled/disabled. If you want to use the IP list, select Enable. ***1**
IP List Mode	・BlackList ・WhiteList	WhiteList	The selected option takes effect, if the “IP List Status” is “Enable”. ***1**
IP Addresses	Enter Valid IP addresses between 0.0.0.0 ~ 255.255.255.255, and each IP address should be separated with commas.	192.168.1.101	Enter one or more IP addresses to register in the IP List. ***1**

***1: For details, see " 9.2. Appendix B: IP List Related Parameters" section.**

9.2 Appendix B: IP List Related Parameters

IP List Status, IP List Mode, and IP Addresses are IP list-related parameters.
The settings for the "When not using an IP list", "When using an IP list as a WhiteList", and "When using an IP list as a BlackList" configurations are described below.

- When not using an IP list
  IP List Status: Disable
  IP List Mode: Any value (this configuration is invalid because IP List Status is “Disable”)
  IP Addresses: Any value (this configuration is invalid because IP List Status is “Disable”)

- When using an IP list as a WhiteList
  IP List Status: Enable
  IP List Mode: WhiteList
  IP Addresses: Any IP address to register as a WhiteList
  If you are following the procedure explained in section "8. Modbus Communication Using Modbus Demo Application" to confirm the operation, set the IP address to "192.168.1.101".

- When using an IP list as a BlackList
  IP List Status: Enable
  IP List Mode: BlackList
  IP Addresses: Any IP address to register as a BlackList
  If you are following the procedure explained in section "8. Modbus Communication Using Modbus Demo Application" to confirm the operation, set the IP address other than "192.168.1.101".

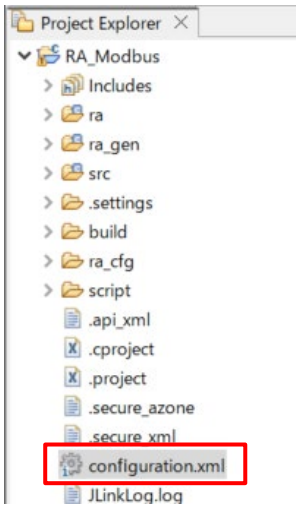
If multiple IP addresses need to be registered in the IP Addresses parameter, separate the IP addresses with a comma (”,”) as shown below.

Example: 192.168.1.101,192.168.1.102,192.168.1.103

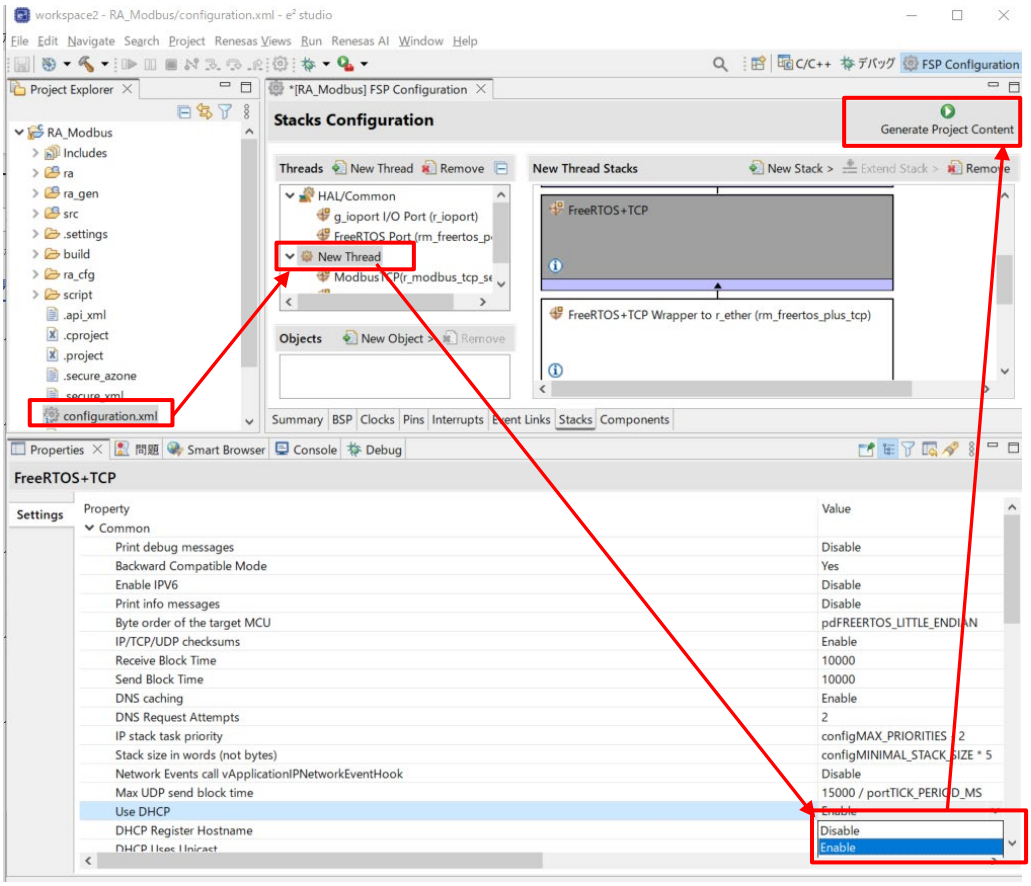
9.3 Appendix C: DHCP Mode

When the operation needs to be confirmed using DHCP mode, execute the procedure below:

1. Open configuration.xml from your Modbus project.



2. Click the "Stacks" tab to open the stack configuration screen and select "FreeRTOS + TCP" in the thread window on the left.
3. Open the properties, change "Use DHCP" to "Enable", and click the "Generate Project Content" button.



9.4 Appendix D: User-defined Function

This section describes the Modbus sample application. Users can register their own implementation of Modbus function code with the Modbus protocol stack.

9.4.1 Register Function Code

Definition file: src/modbus_func.c
Define the function to be registered in the callback function of the Modbus protocol stack.

9.4.2 User-defined Functions

User-defined functions are defined in the src/modbus_user.c
User-defined Read/Write functions are used to process each function.

Some functions access the following components of the evaluation board.

Evaluation Boards	User LED			User-Switch	
	①	②	③	①	②
EK-RA6M3	User LED1	User LED2	User LED3	User-Switch S1	User-Switch S2
EK-RA6M4	User LED1	User LED2	User LED3	User-Switch S1	User-Switch S2
EK-RA6M5	User LED1	User LED2	User LED3	User-Switch S1	User-Switch S2
EK-RA8D1	User LED1	User LED2	User LED3	User-Switch S1	User-Switch S2
EK-RA8M1	User LED1	User LED2	User LED3	User-Switch S1	User-Switch S2
MCK-RA8T1	LED1	LED2	LED3	-	-

Read/Write functions and tables are provided that correspond to each address of the coil/discrete input/holding register/input register.
The evaluation board parts and global variables accessed by the functions in each table are as follows.

【Read Coils】	
address	**Access**
0001	Above table User - LED①, g_coils_area
0002	Above table User - LED②, g_coils_area
0003	Above table User - LED③, g_coils_area
0004	g_coils_area
0005	g_coils_area
0006	g_coils_area
0007	g_coils_area
0008	g_coils_area

【Write_Single_Coils】	
address	**access**
0001	Above table User - LED①, g_coils_area *1
0002	Above table User - LED②, g_coils_area *1
0003	Above table User - LED③, g_coils_area *1
0004	g_coils_area
0005	g_coils_area
0006	g_coils_area
0007	g_coils_area
0008	g_coils_area

*1：Each LED on the evaluation board will turn on/off depending on the value written by this function.

【Read_Discrete_Inputs】

address	access
1001	g_discrete_input_area
1002	g_discrete_input_area
1003	g_discrete_input_area
1004	g_discrete_input_area
1005	g_discrete_input_area
1006	Above table User-Switch①, g_discrete_input_area *2
1007	Above table User-Switch②, g_discrete_input_area *2
1008	g_discrete_input_area
1009	g_discrete_input_area
10010	ILLEGAL DATA ADDRESS
10011	g_discrete_input_area
10012	g_discrete_input_area

*2：The value read by this function is displayed in “Discrete Inputs” of the Modbus demo application.

【Read_Discrete_Inputs】

address	access
3001	g_input_reg_area
3002	g_input_reg_area
3003	g_input_reg_area
3004	ILLEGAL DATA ADDRESS
3005	ILLEGAL DATA ADDRESS
3006	ILLEGAL DATA ADDRESS
3007	ILLEGAL DATA ADDRESS
3008	g_input_reg_area

【READ_HOLDING_REGISTERS】

address	access
4001	g_holding_reg_area
4002	g_holding_reg_area
4003	g_holding_reg_area
4004	ILLEGAL DATA ADDRESS
4005	ILLEGAL DATA ADDRESS
4006	ILLEGAL DATA ADDRESS
4007	g_holding_reg_area

【WRITE_SINGLE_REGISTER】

address	access
4001	g_holding_reg_area
4002	g_holding_reg_area
4003	g_holding_reg_area
4004	ILLEGAL DATA ADDRESS
4005	ILLEGAL DATA ADDRESS
4006	ILLEGAL DATA ADDRESS
4007	g_holding_reg_area

9.5 Appendix E: Multiple Client Communication

The Modbus protocol stack can connect up to three clients using Modbus communication.
If you create a project using the steps in "**6. Setting Up the Modbus Sample Project**", you can communicate continuously with the client under the following conditions.

- Number of clients: 3
- Communication interval: 1000ms
- Communication Timeout Time：3000ms
- Do not connect devices that are not related to Modbus communication in the same network

If you want the communication interval or timeout time to be shorter than the above time, or if you need to connect other devices within the same network, change the following:

- Go to “Total number of available network buffers” property of “Stacks” → “FreeRTOS+TCP” → “Common “and increase its value.
  Note that the total number of available network buffers uses 56 bytes of RAM, and the amount of RAM used will increase proportionally to this value.
  (If the total number of available network buffers is 30, the amount of RAM used will be 56 x 30 = 1680 bytes)
- **Below is the settings for EK-RA6M3 / EK-RA6M4 / EK-RA6M5 / EK-RA8D1 / EK-RA8M1 / MCK-RA8T1.**
  Go to "Stacks" → "g_ether0 Ethernet" → "Module g_ether0 Ethernet (r_ether)" → "Buffers" → “Number of RX buffer" and change its value greater than or equal to 1 (default value). The recommended value is [Number of clients].
  (The RAM size used by the number of RX buffers is 1536 bytes.)
  (If you change the Number of RX buffer to 3, the total is 1536 x 3 = 4608 bytes)

Note: If you want to use the Modbus sample project in your system, evaluate it thoroughly.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jun.27.25	-	First version
1.10	Sep.25.25		**Added MCK-RA8T2**
		Page 4	Updated “Table 1.2”
		Page 6	Updated “3.1 Modbus Sample Project”
		Page 9	Added “Figure 5.3”
		Page 18 - 43	“6.2 Creation of new Modbus Project” is divided into common procedures for all evaluation boards and procedures for each evaluation board, and procedures for MCK-RA8T2 are added
			Updated Operating Environment Requirements
		Page 7	Updated “Table 4.1”
			Added description of multi-client communication
		Page 56	Added “9.5 Appendix E: Multiple Client Communication”
1.20	Oct.31.25		**Added EK-RA8D2, EK-RA8P1 and EK-RA8M2**
		Page 4	Updated “Table 1.2”
		Page 6	Updated “3.1 Modbus Sample Project”
		Page 9	Updated “Figure 5.3”
		Page 16	Changed the title of "6.1.2" to " MCK-RA8T2 / EK-RA8D2 / EK-RA8P1 / EK-RA8M2 Importing Procedures"
		Page 45 – 67	Added "6.2.4 EK-RA8D2 Importing Procedures", "6.2.5 EK-RA8P1 Importing Procedures" and "6.2.6 EK-RA8M2 Importing Procedures"
		Page 80	Changed “9.5 Appendix E: Multiple Client Communication”
			Updated Operating Environment Requirements
		Page 7	Updated “Table 4.1”
			Additional Procedures for Upgrading FSP Version
		Page 38	Added step 3 to "6.2.3"

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)
   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on
   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state
   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins
   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals
   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin
   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses
   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products
   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.